Introdução ao Aprendizado de Máquina:

Redes Artificiais Neurais (RNA)

Artificial Neural Networks (ANN)

Thales A. P. West, Ph.D.

Aula: Duas Partes

Parte 1. O que são e como funcionam as RNA?

Parte 2. Como estimar parâmetros em RNA?

Aula: Duas Partes

Parte 1. O que são e como funcionam as RNA?

Parte 2. Como estimar parâmetros em RNA?

Como é uma Rede Neural Artificial (RNA)?

BULLETIN OF
MATHEMATICAL BIOPHYSICS
VOLUME 5, 1943

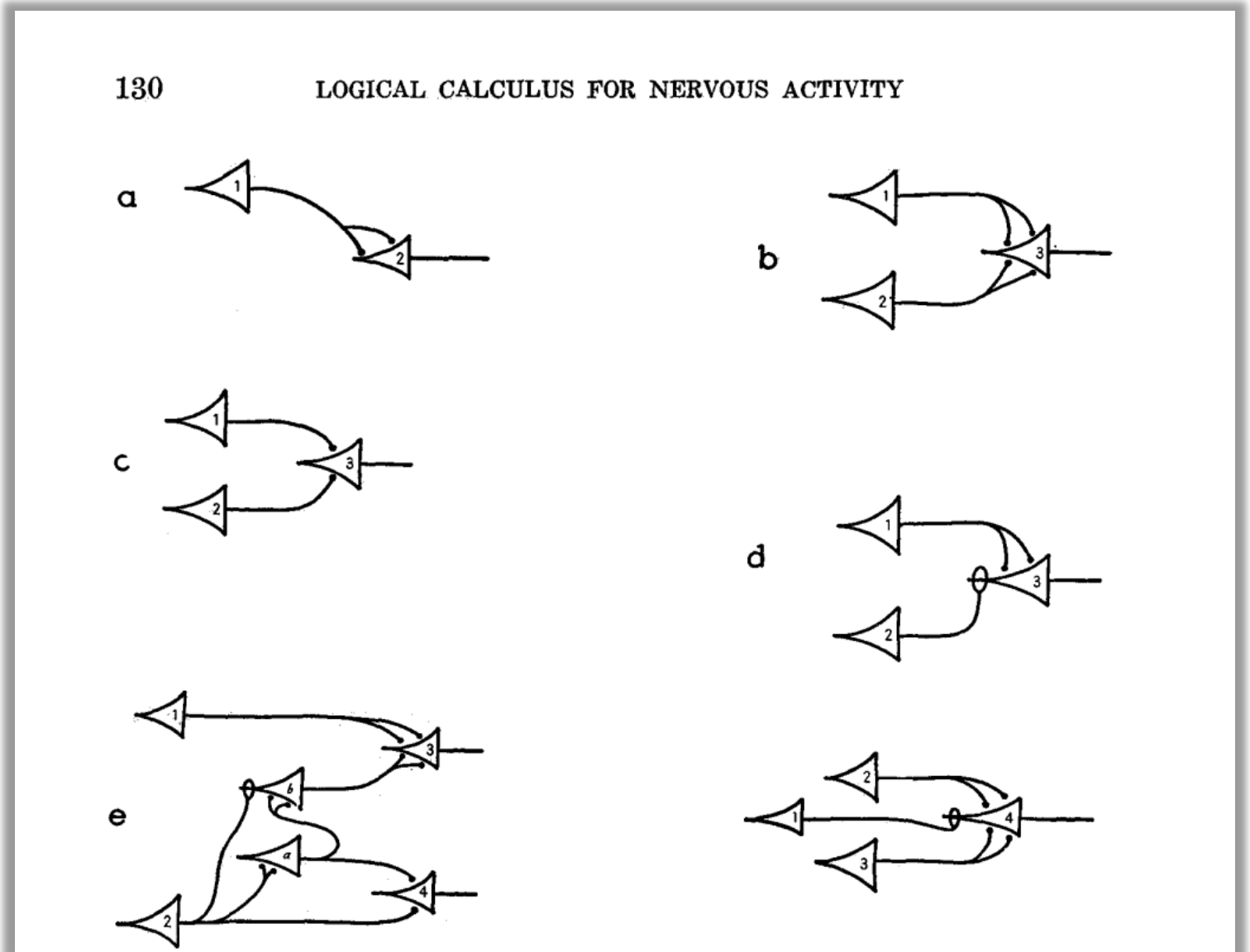
A LOGICAL CALCULUS OF THE IDEAS IMMANENT IN NERVOUS ACTIVITY

WARREN S. MCCULLOCH AND WALTER PITTS

FROM THE UNIVERSITY OF ILLINOIS, COLLEGE OF MEDICINE,
DEPARTMENT OF PSYCHIATRY AT THE ILLINOIS NEUROPSYCHIATRIC INSTITUTE,
AND THE UNIVERSITY OF CHICAGO

Because of the "all-or-none" character of nervous activity, neural events and the relations among them can be treated by means of propositional logic. It is found that the behavior of every net can be described in these terms, with the addition of more complicated logical means for nets containing circles; and that for any logical expression satisfying certain conditions, one can find a net behaving in the fashion it describes. It is shown that many particular choices among possible neurophysiological assumptions are equivalent, in the sense that for every net behaving under one assumption, there exists another net which behaves under the other and gives the same results, although perhaps not in the same time. Various applications of the calculus are discussed.

Como é uma Rede Neural Artificial (RNA)?


130 LOGICAL CALCULUS FOR NERVOUS ACTIVITY

a

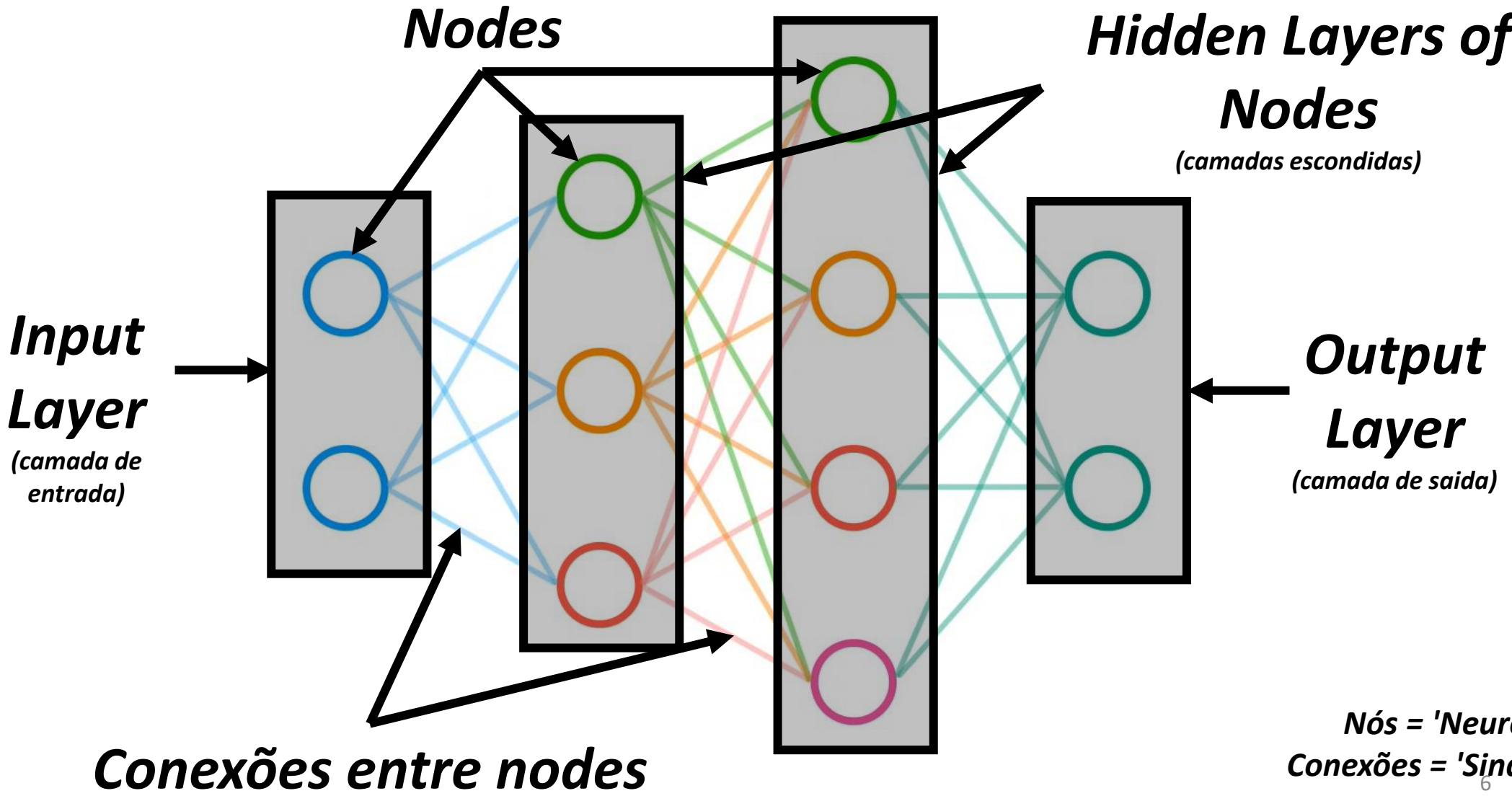
b

c

d

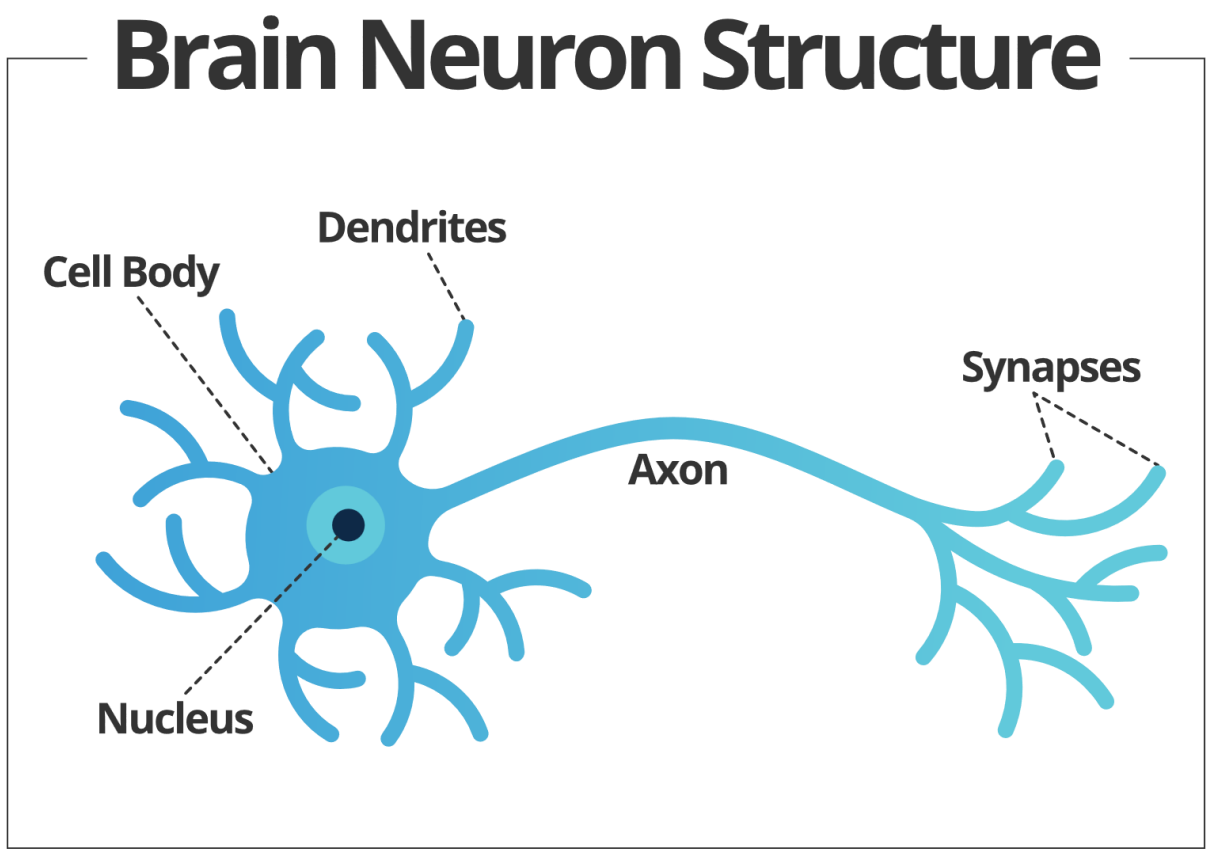
e

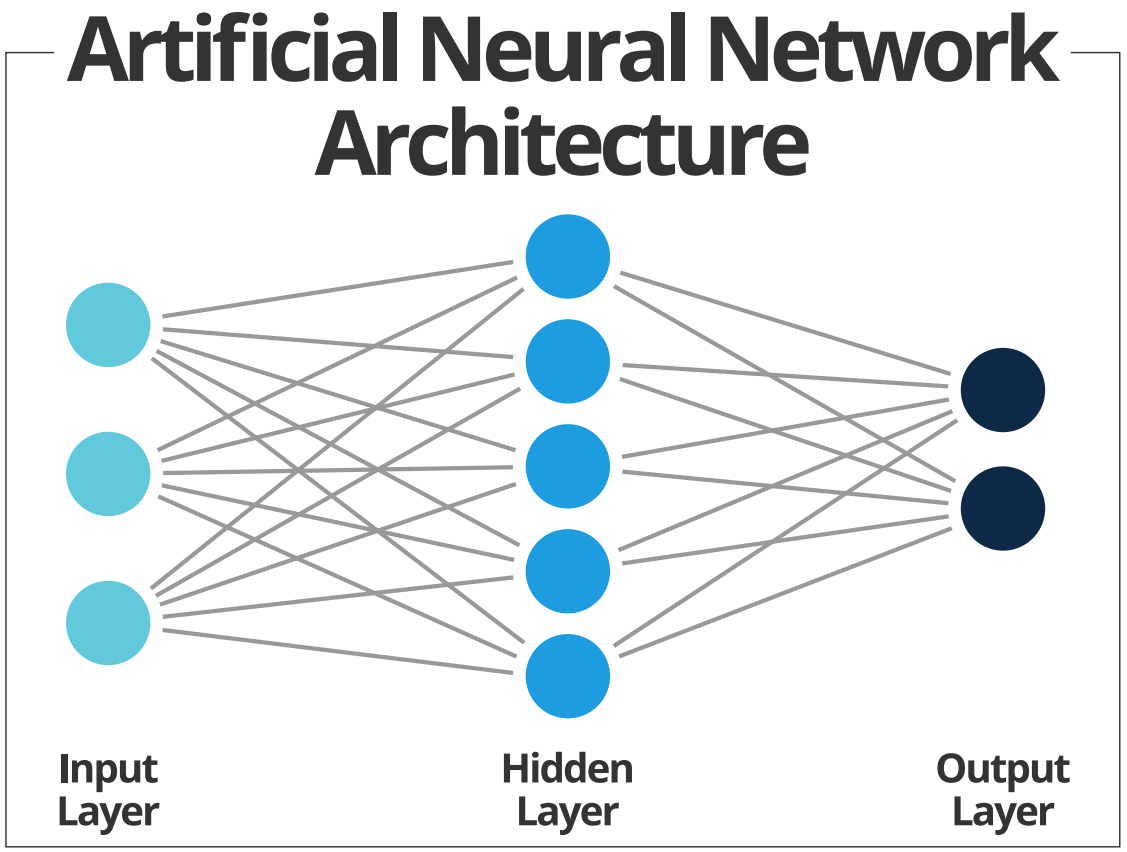
Como é uma Rede Neural Artificial (RNA)?



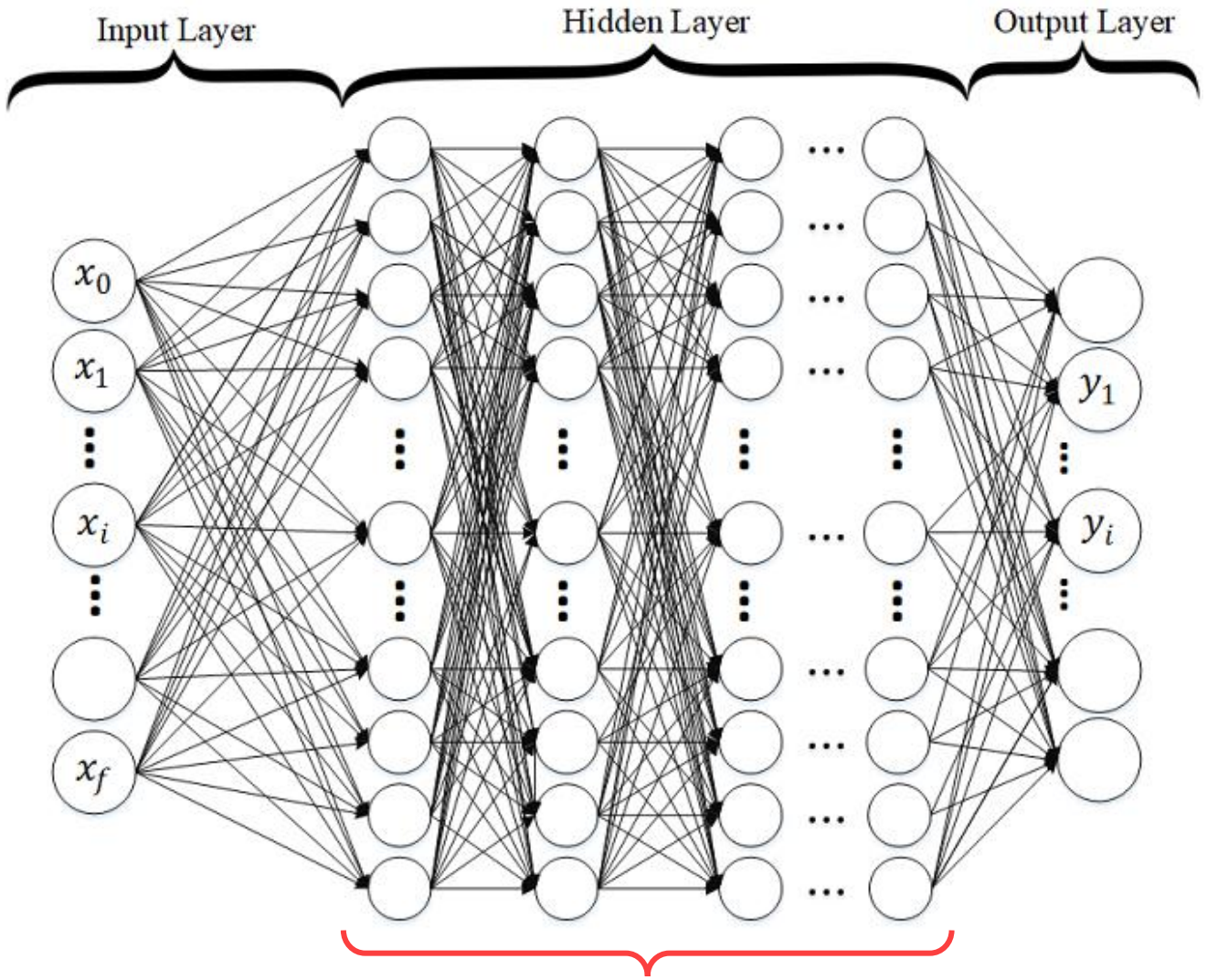
Nós = 'Neurônios'
Conexões = 'Sinapses'

Como é uma Rede Neural Artificial (RNA)?





Como é uma Rede Neural Artificial (RNA)?



Normalmente, quando a RNA tem mais de 3 camadas ocultas, usamos o termo “deep learning"

Aplicações de modelos de RNA

ELSEVIER

Journal of Cleaner Production
Volume 225, 10 July 2019, Pages 833-856

Modelling carbon emission intensity: Application of artificial neural network

Methods in Ecology and Evolution | BRITISH ECOLOGICAL SOCIETY

RESEARCH ARTICLE | Open Access

Using deep convolutional neural networks to forecast spatial patterns of Amazonian deforestation

ELSEVIER

Computers and Electronics in Agriculture
Volume 166, November 2019, 105031

Using boosted tree regression and artificial neural networks to forecast upland rice yield under climate change in Sahel

LIMNOLOGY AND OCEANOGRAPHY | ASLO Association for the Sciences of Limnology and Oceanography

Article | Free Access

Stream hydrological and ecological responses to climate change assessed with an artificial neural network

Geophysical Research Letters

Research Letter | Open Access

Using Machine Learning to Analyze Physical Causes of Climate Change: A Case Study of U.S. Midwest Extreme Precipitation

ELSEVIER

Solar Energy
Volume 180, 1 March 2019, Pages 622-639

Review

Modeling of solar energy systems using artificial neural network: A comprehensive review

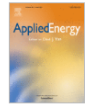
ELSEVIER

CATENA
Volume 186, March 2020, 104394

A remote sensing and artificial neural network-based integrated agricultural drought index: Index development and applications

ELSEVIER

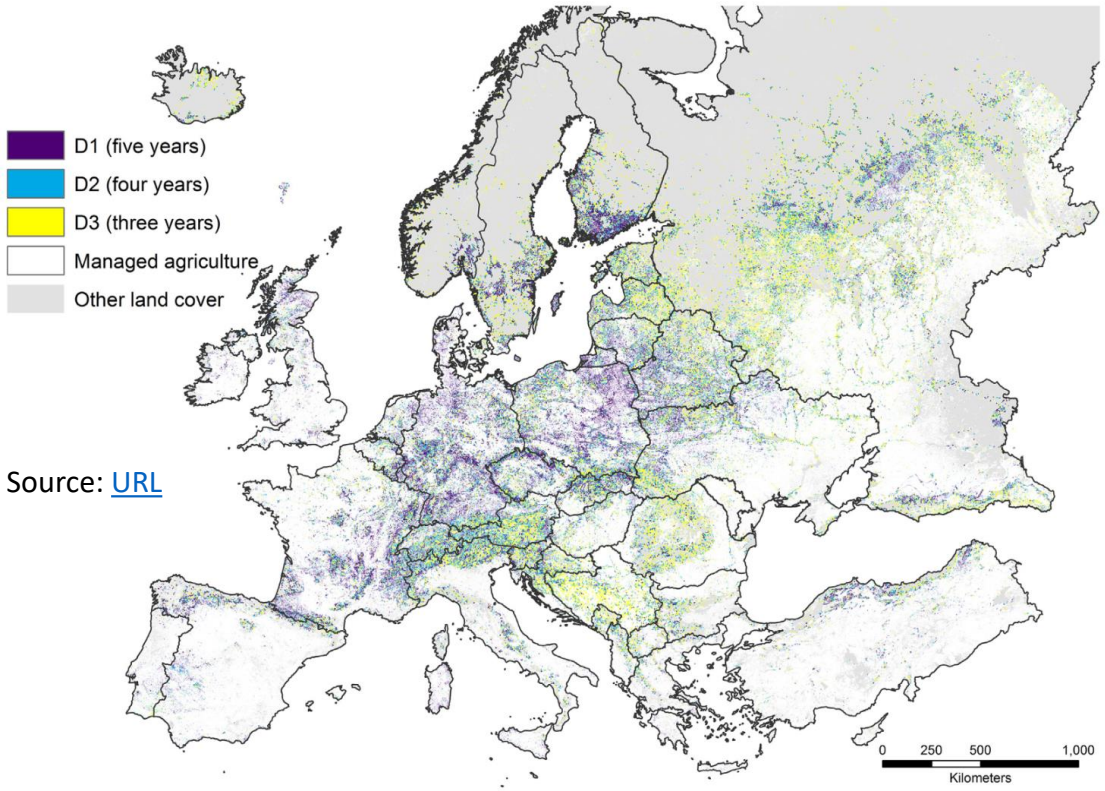
Applied Energy
Volume 291, 1 June 2021, 116818



GIS aided sustainable urban road management with a unifying queueing and neural network model

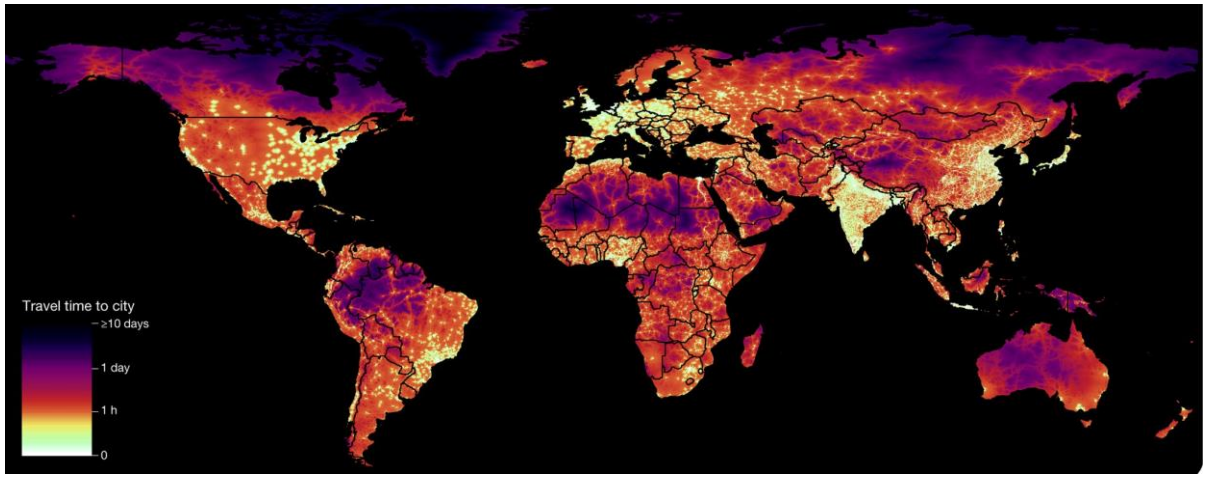
Como funciona uma Rede Neural Artificial (RNA)?

Podemos prever o abandono de terras com base na proximidade das cidades?



Source: URL

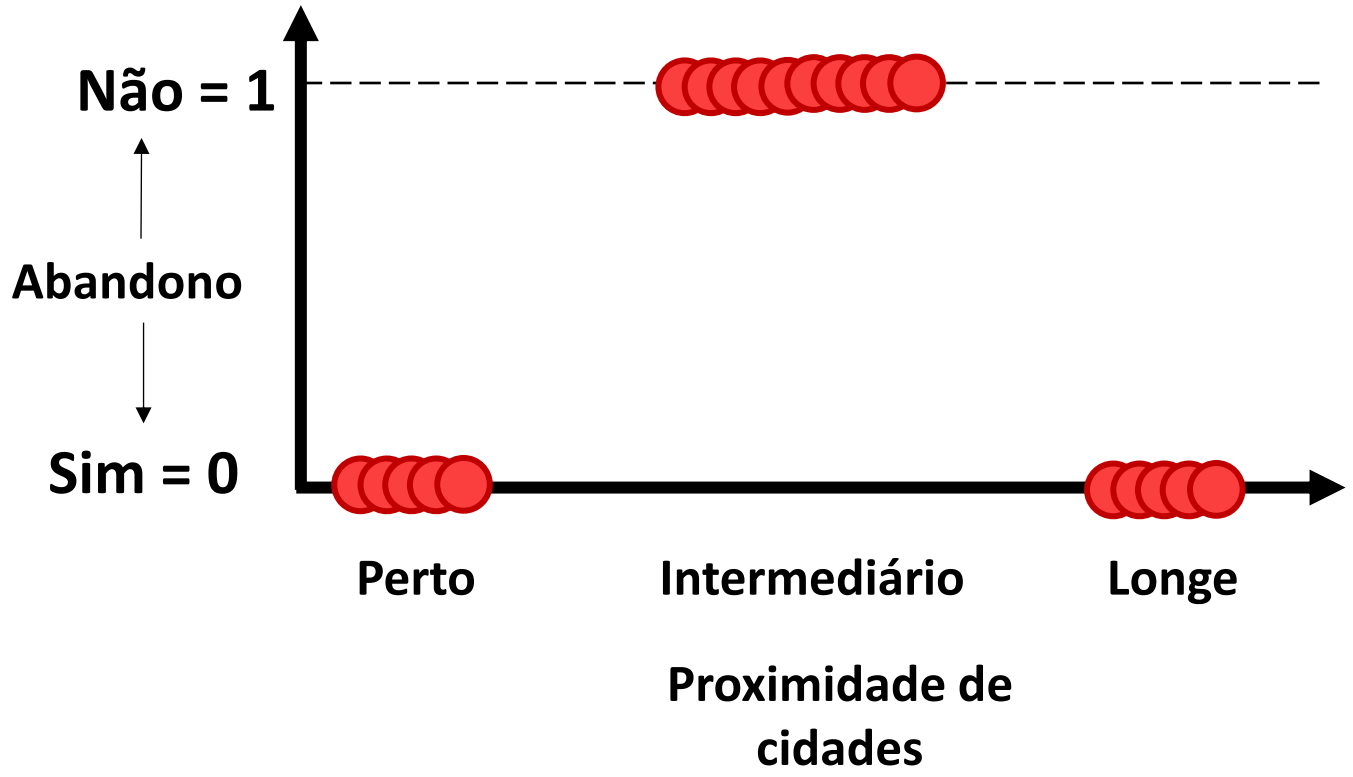
Fig. 2. Study area with maps of agricultural abandonment defined by the three definitions (Estel et al., 2015). Abandonment locations were magnified to increase visibility.



https://www.nature.com/articles/nature25181

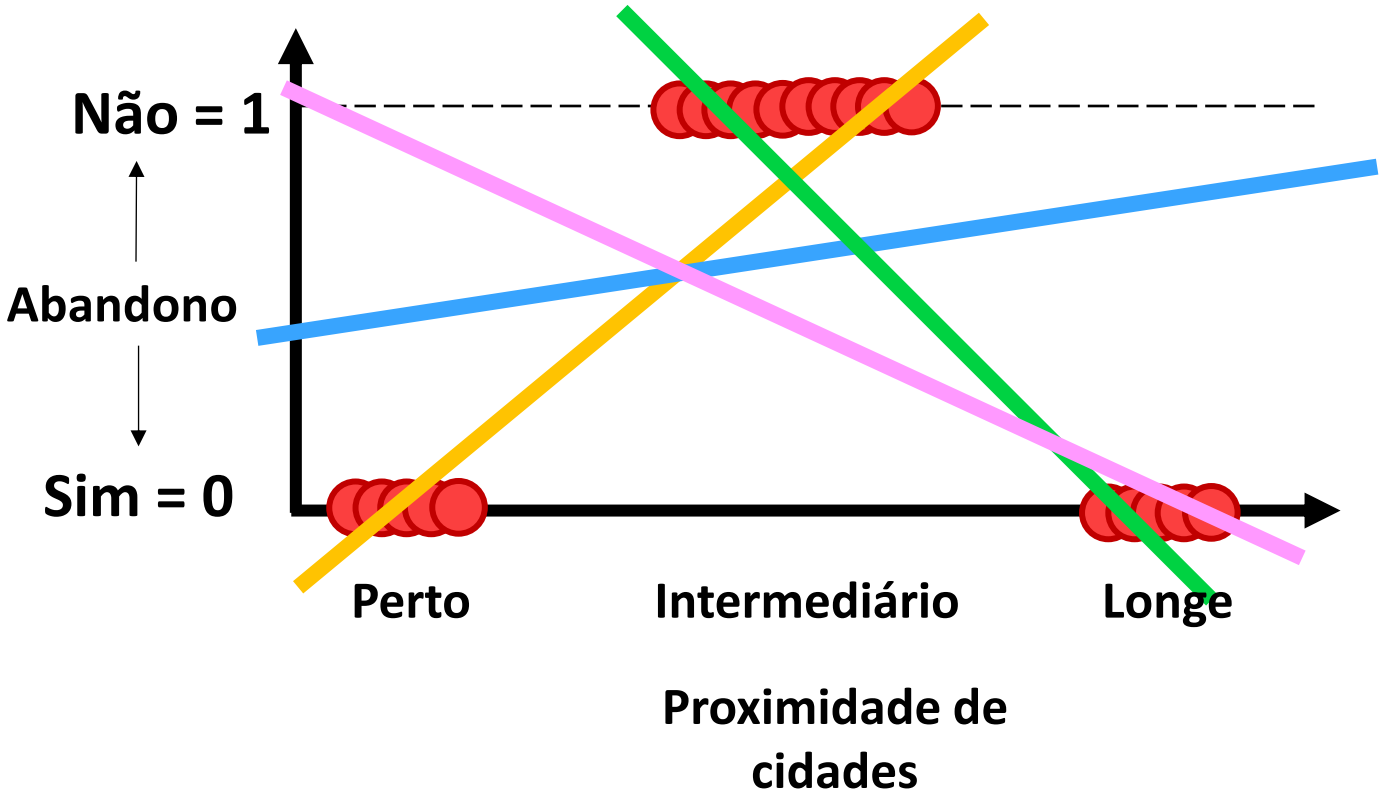
Como funciona uma Rede Neural Artificial (RNA)?

Podemos prever o abandono de terras com base na proximidade das cidades?



Como funciona uma Rede Neural Artificial (RNA)?

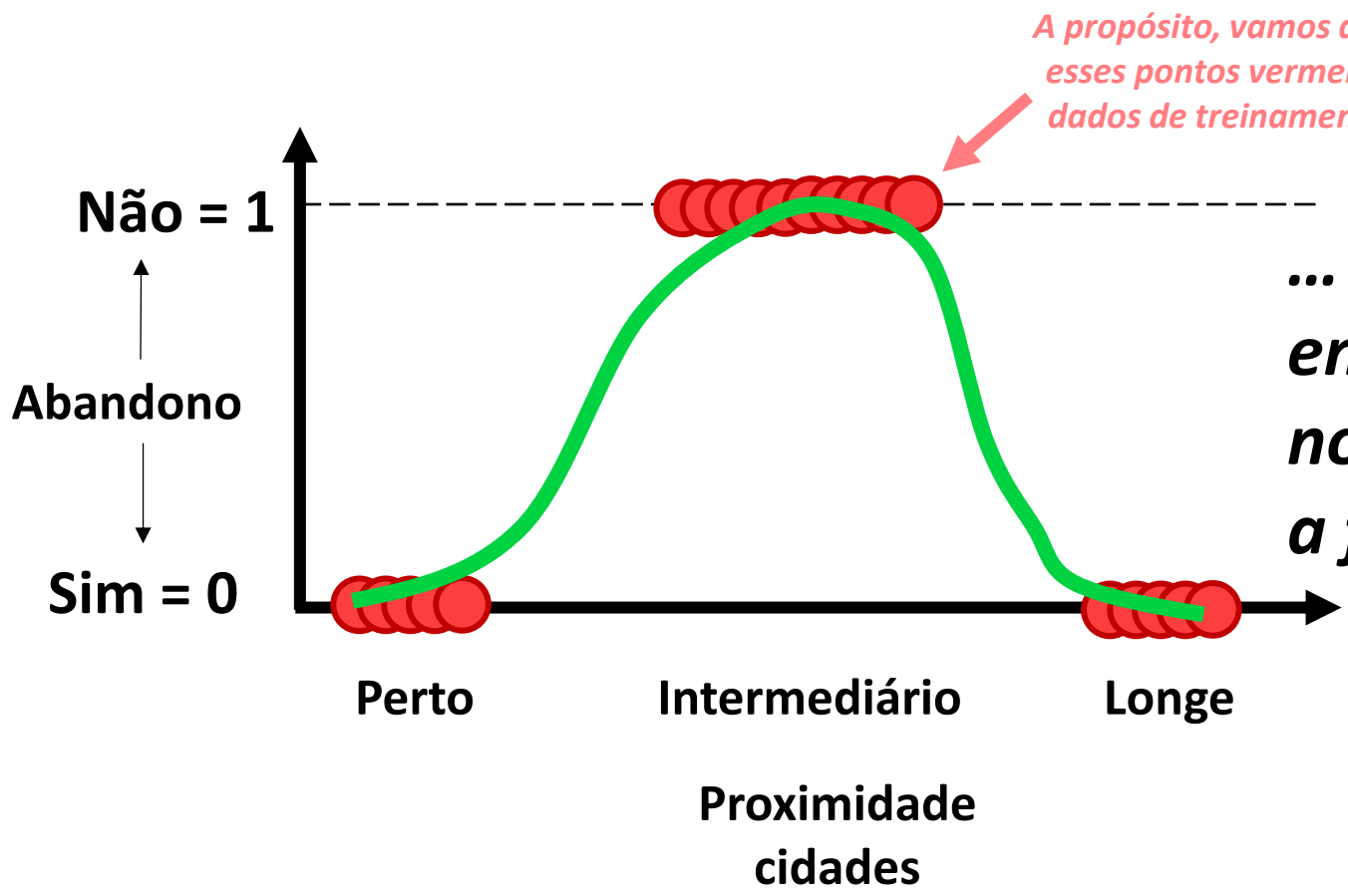
Podemos prever o abandono de terras com base na proximidade das cidades?



Uma linha reta (de um modelo de regressão linear) não nos ajudará a fazer previsões!

Como funciona uma Rede Neural Artificial (RNA)?

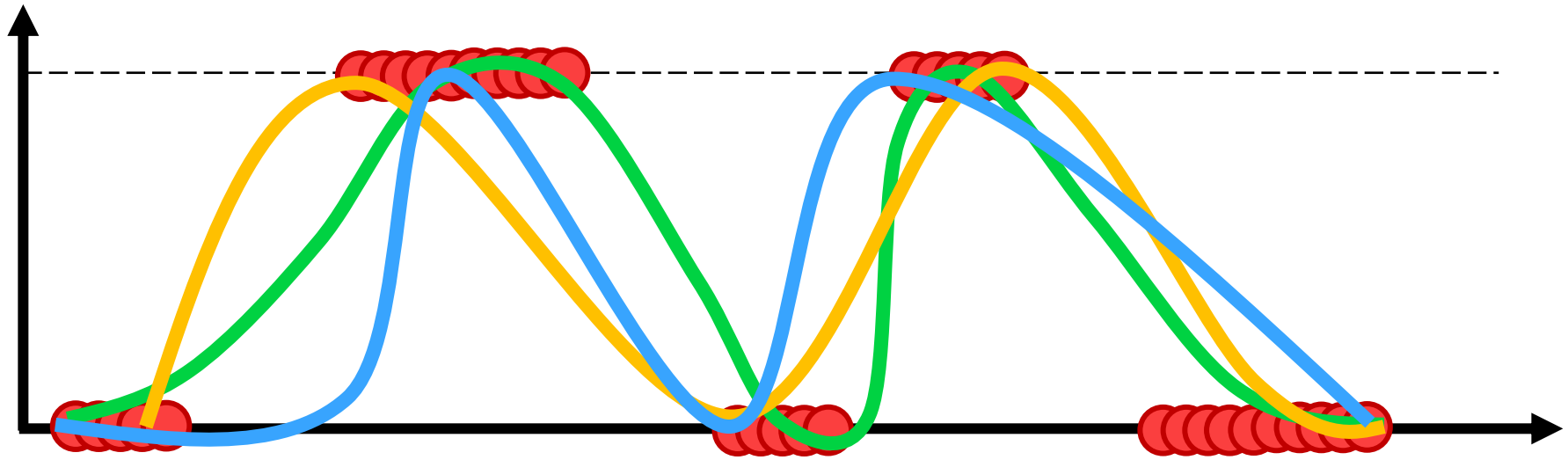
Podemos prever o abandono de terras com base na proximidade das cidades?



… mas uma RNA pode encaixar um "rabisco" nos dados e nos ajudar a fazer previsões!

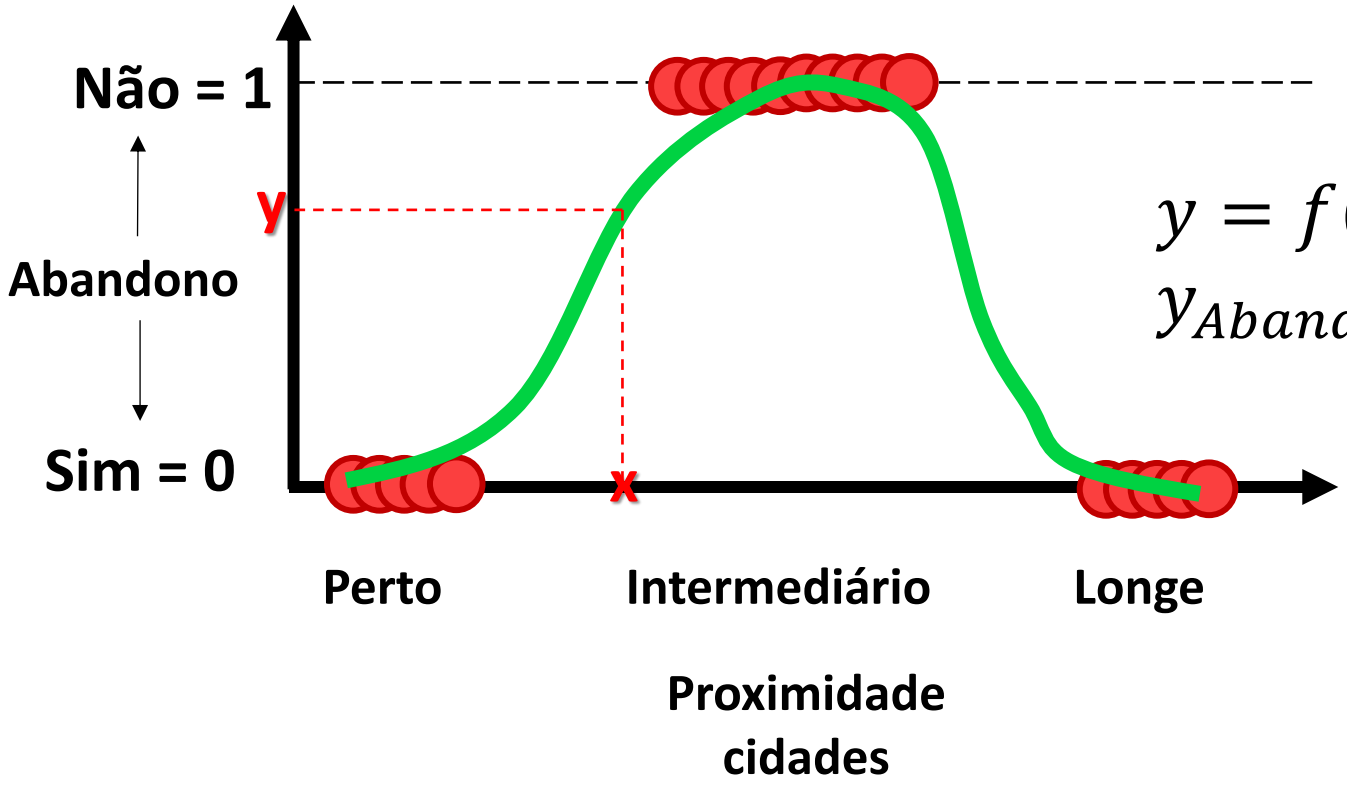
Como funciona uma Rede Neural Artificial (RNA)?

Os "rabiscos" de RNAs podem se encaixar em padrões complexos (não lineares) e podem ser muito úteis em varios casos!



Como funciona uma Rede Neural Artificial (RNA)?

Podemos prever o abandono de terras com base na proximidade das cidades?

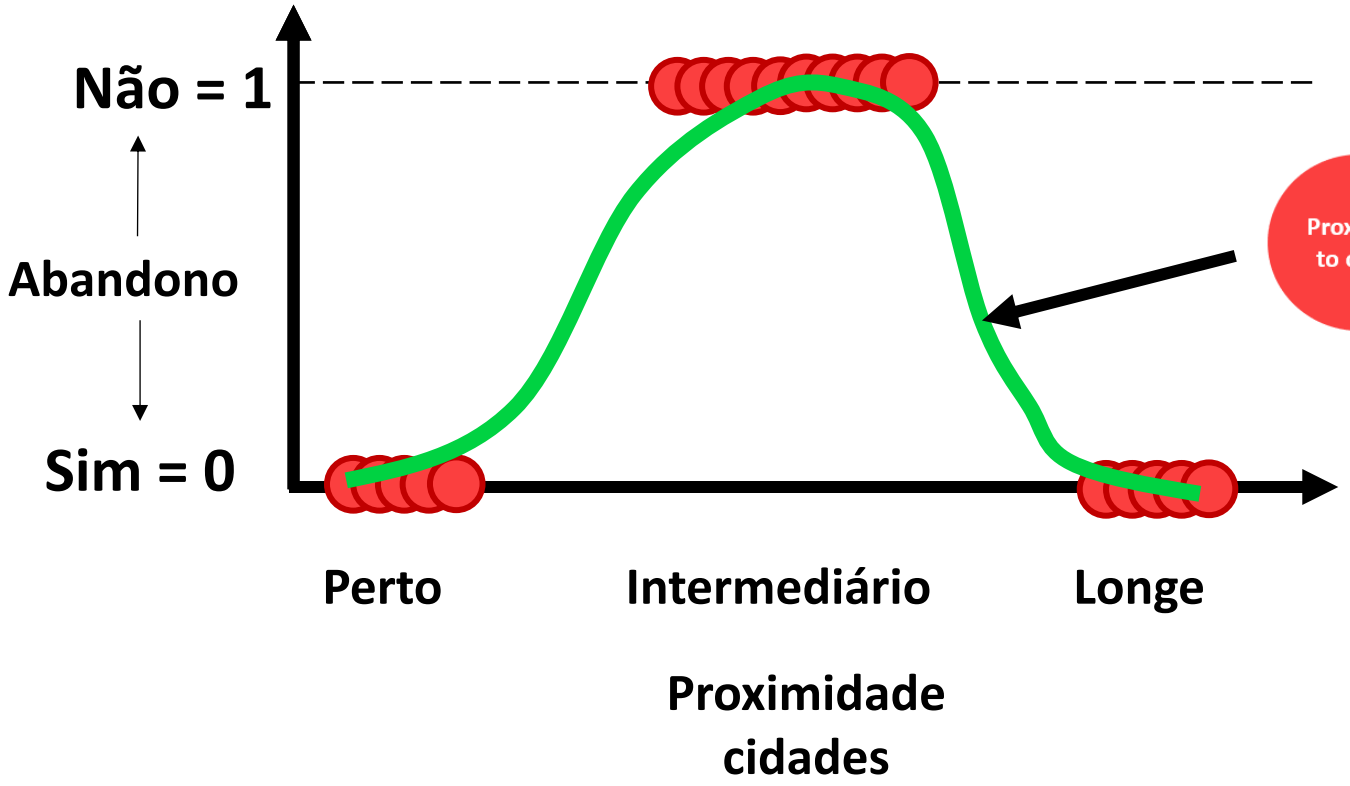


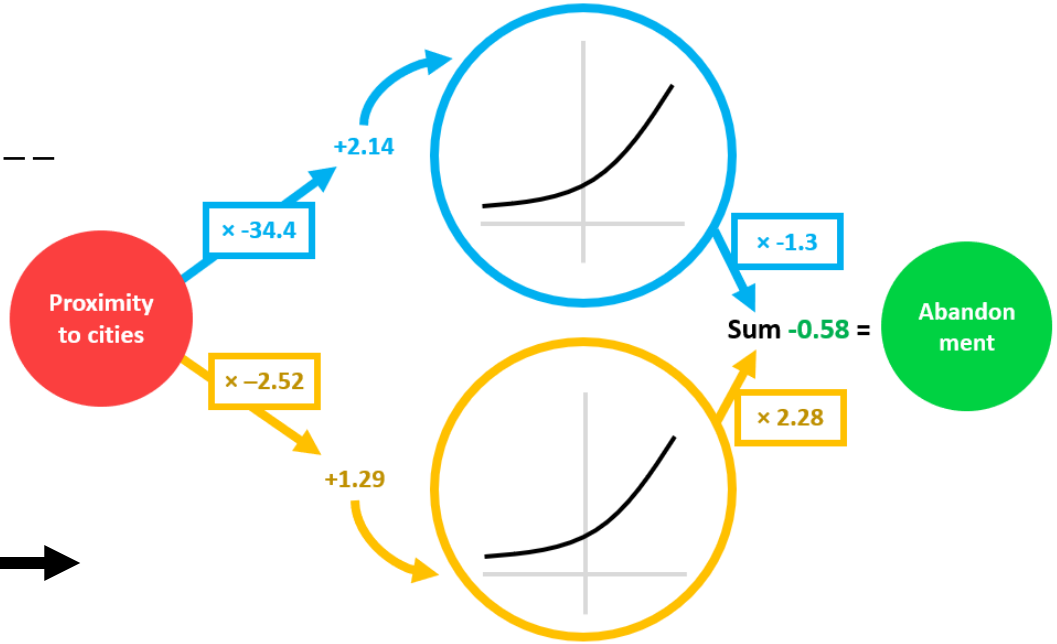
$$y = f(x)$$

$$y_{Abandono} = f(x_{Prox.\,cidades})$$

Como funciona uma Rede Neural Artificial (RNA)?

Podemos prever o abandono de terras com base na proximidade das cidades?

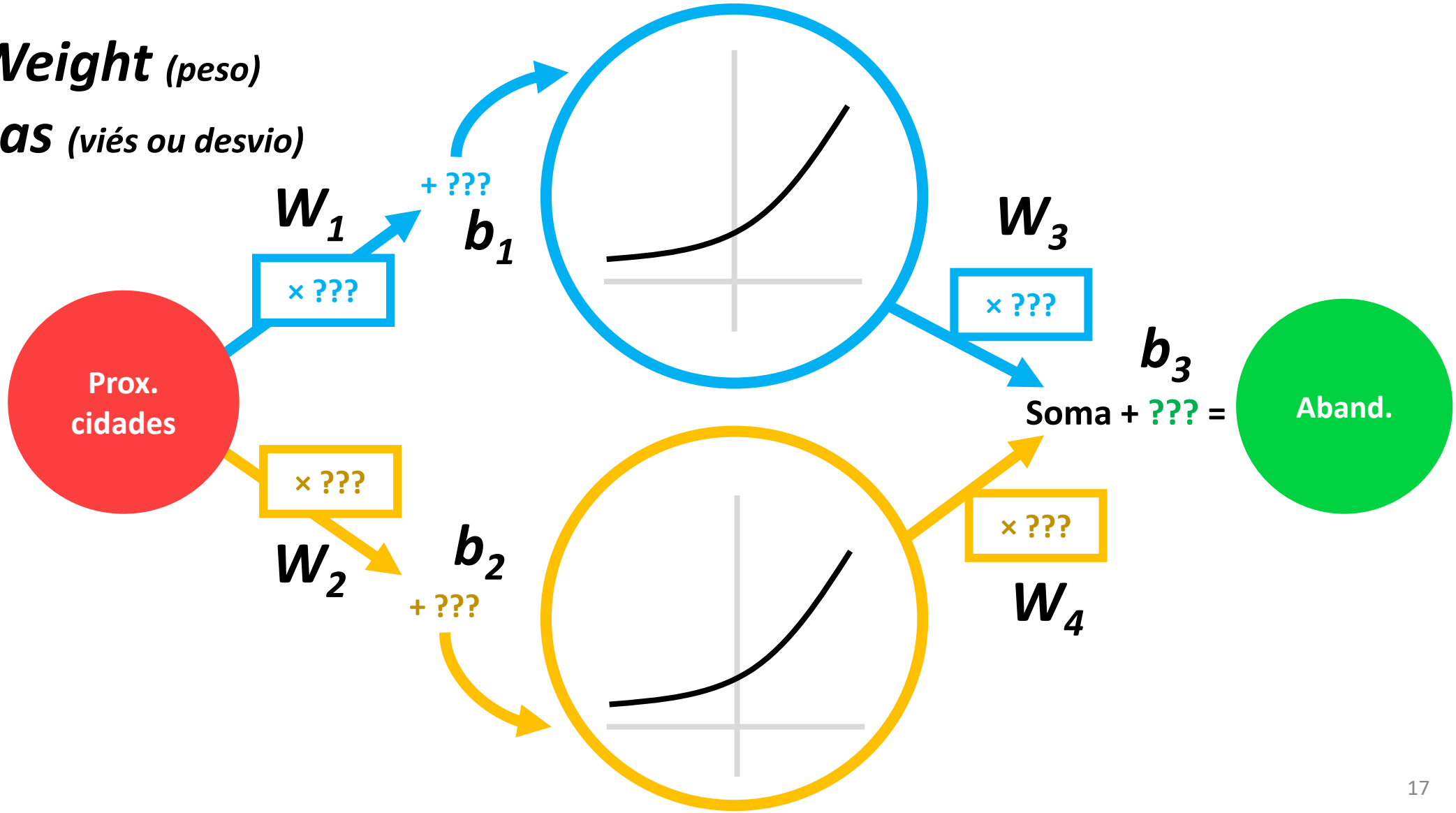




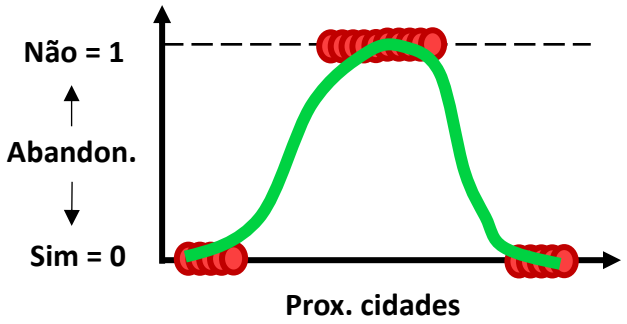
Como funciona uma Rede Neural Artificial (RNA)?

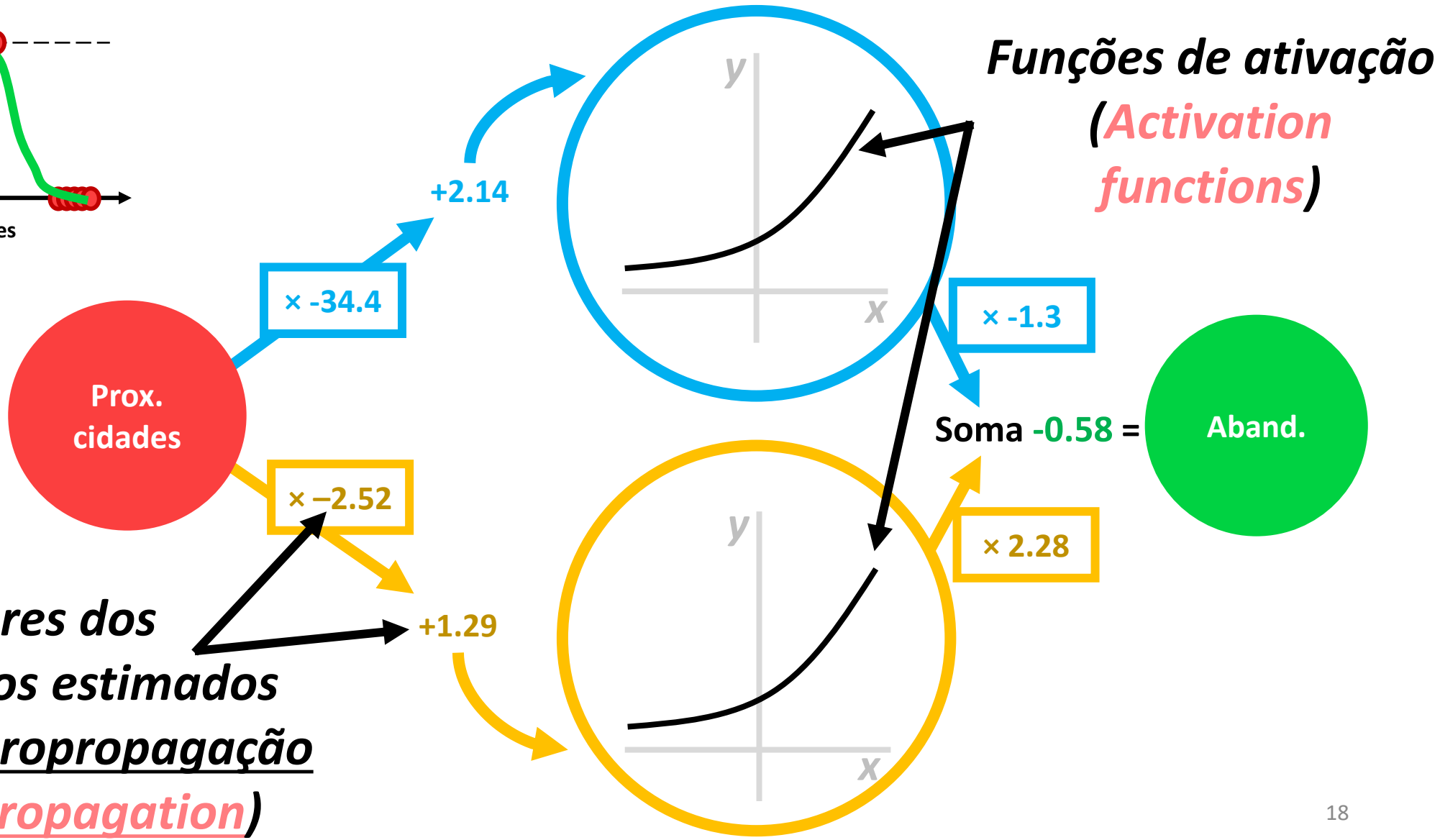
***W = Weight** (peso)*

***b = Bias** (viés ou desvio)*



Como funciona uma Rede Neural Artificial (RNA)?

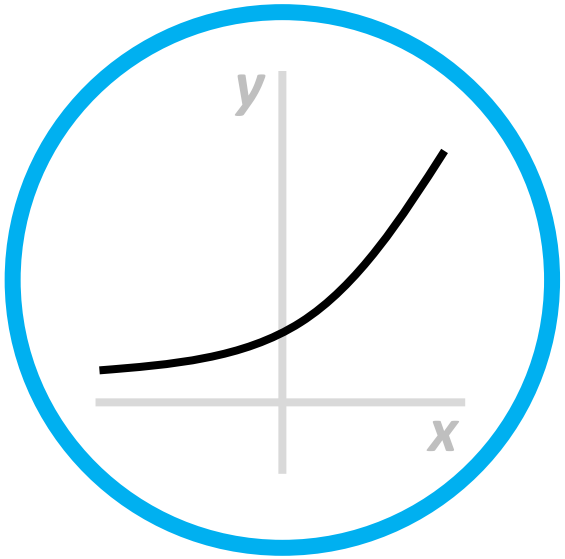




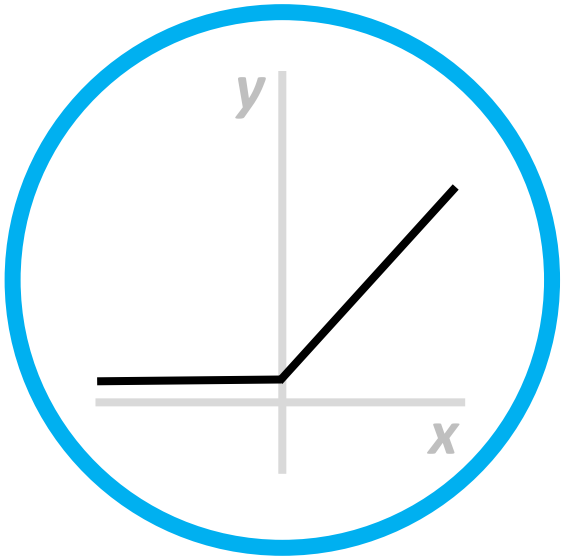
Funções de ativação (Activation functions)

Valores dos parâmetros estimados (usando retropropagação ou backpropagation)

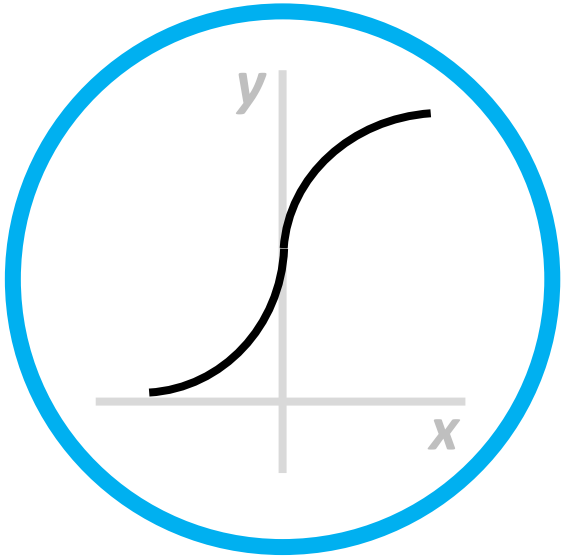
Alguns tipos de Funções de Ativação



Softplus



Rectified Linear Unit (ReLU)

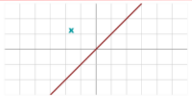
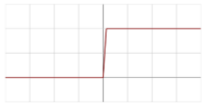
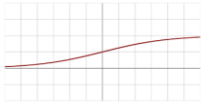
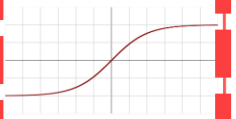
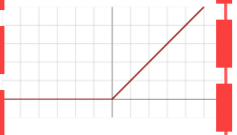
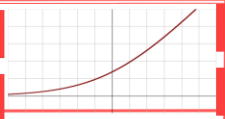
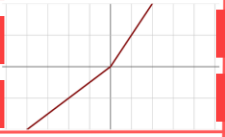
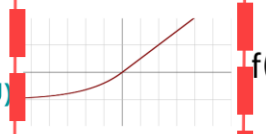


Sigmoid

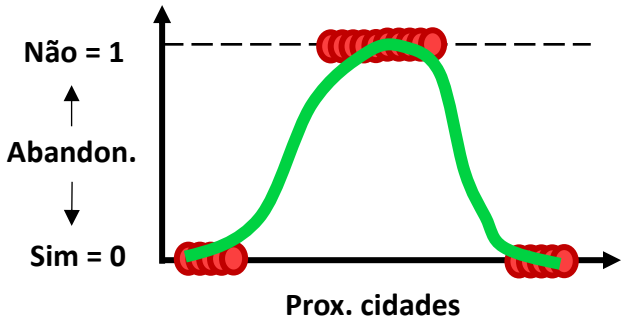
Alguns tipos de Funções de Ativação

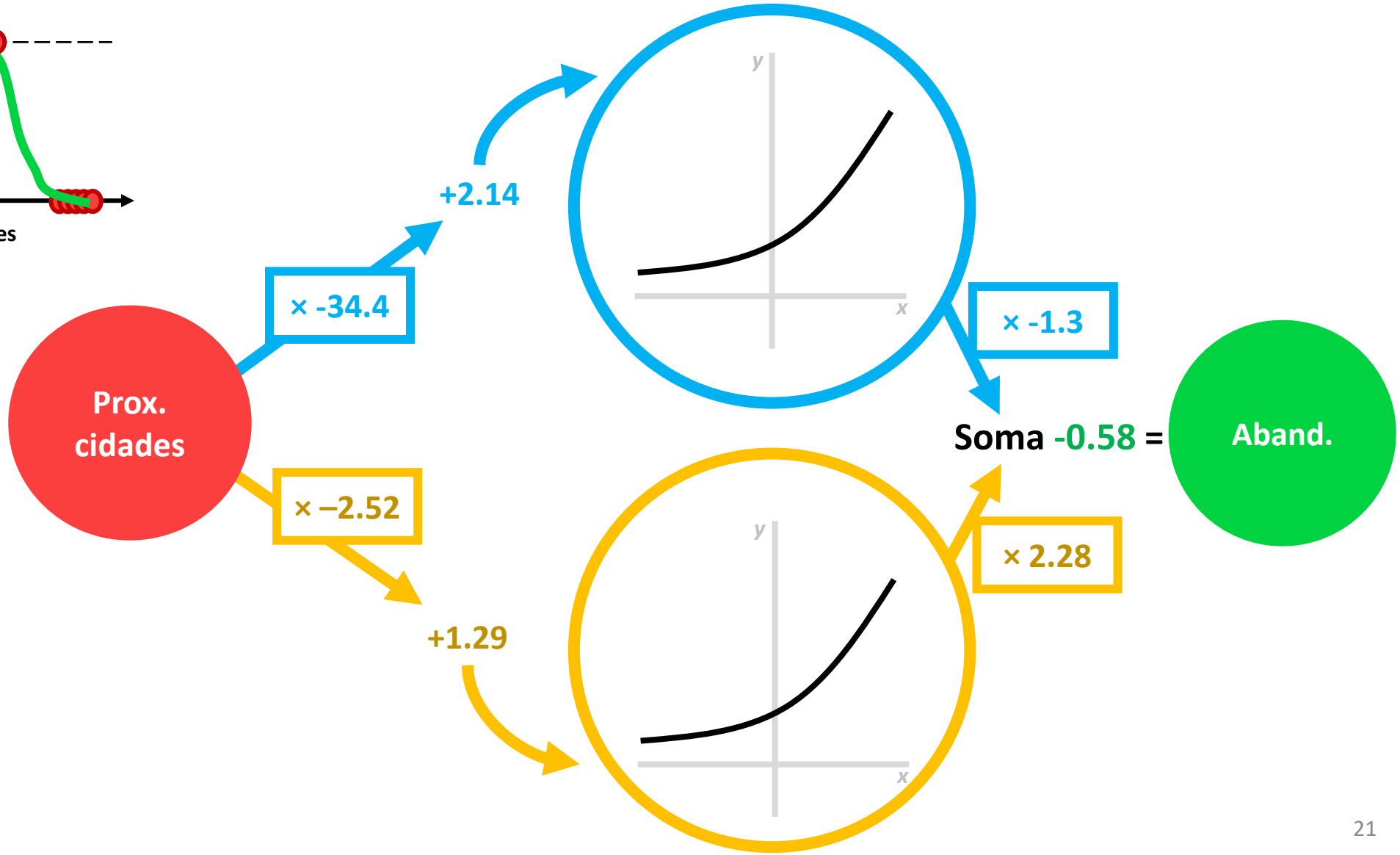
Escolher a função de ativação correta para RNA é uma decisão importante, pois influencia o desempenho do modelo!

Uma mesma RNA pode conter diferentes funções de ativação em camadas diferentes...!

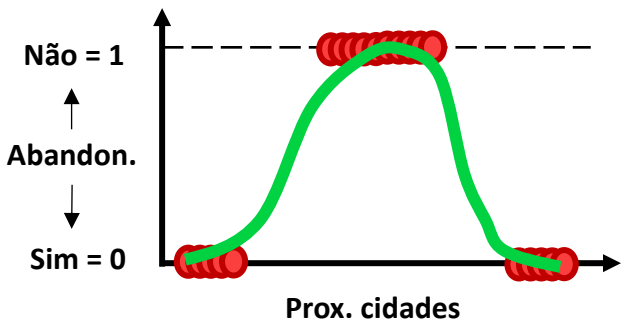
ACTIVATION FUNCTION	PLOT	EQUATION	DERIVATIVE	RANGE
Linear		$f(x) = x$	$f'(x) = 1$	$(-\infty, \infty)$
Binary Step		$f(x) = \begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{if } x \neq 0 \\ \text{undefined} & \text{if } x = 0 \end{cases}$	$\{0, 1\}$
Sigmoid		$f(x) = \sigma(x) = \frac{1}{1+e^{-x}}$	$f'(x) = f(x)(1 - f(x))$	$(0, 1)$
Hyperbolic Tangent(tanh)		$f(x) = \tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$	$f'(x) = 1 - f(x)^2$	$(-1, 1)$
Rectified Linear Unit(ReLU)		$f(x) = \begin{cases} 0 & \text{if } x<0 \\ x & \text{if } x\geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x > 0 \\ \text{undefined} & \text{if } x = 0 \end{cases}$	$[0, \infty)$
Softplus		$f(x) = \ln(1 + e^{x})$	$f'(x) = \frac{1}{1+e^{-x}}$	$(0, 1)$
Leaky ReLU		$f(x) = \begin{cases} 0.01x & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0.01 & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases}$	$(-1, 1)$
Exponential Linear Unit(ELU)		$f(x) = \begin{cases} \alpha(e^{x} - 1) & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$	$f'(x) = \begin{cases} \alpha e^{x} & \text{if } x < 0 \\ 1 & \text{if } x > 0 \\ 1 & \text{if } x = 0 \text{ and } \alpha = 1 \end{cases}$	$[0, \infty)$

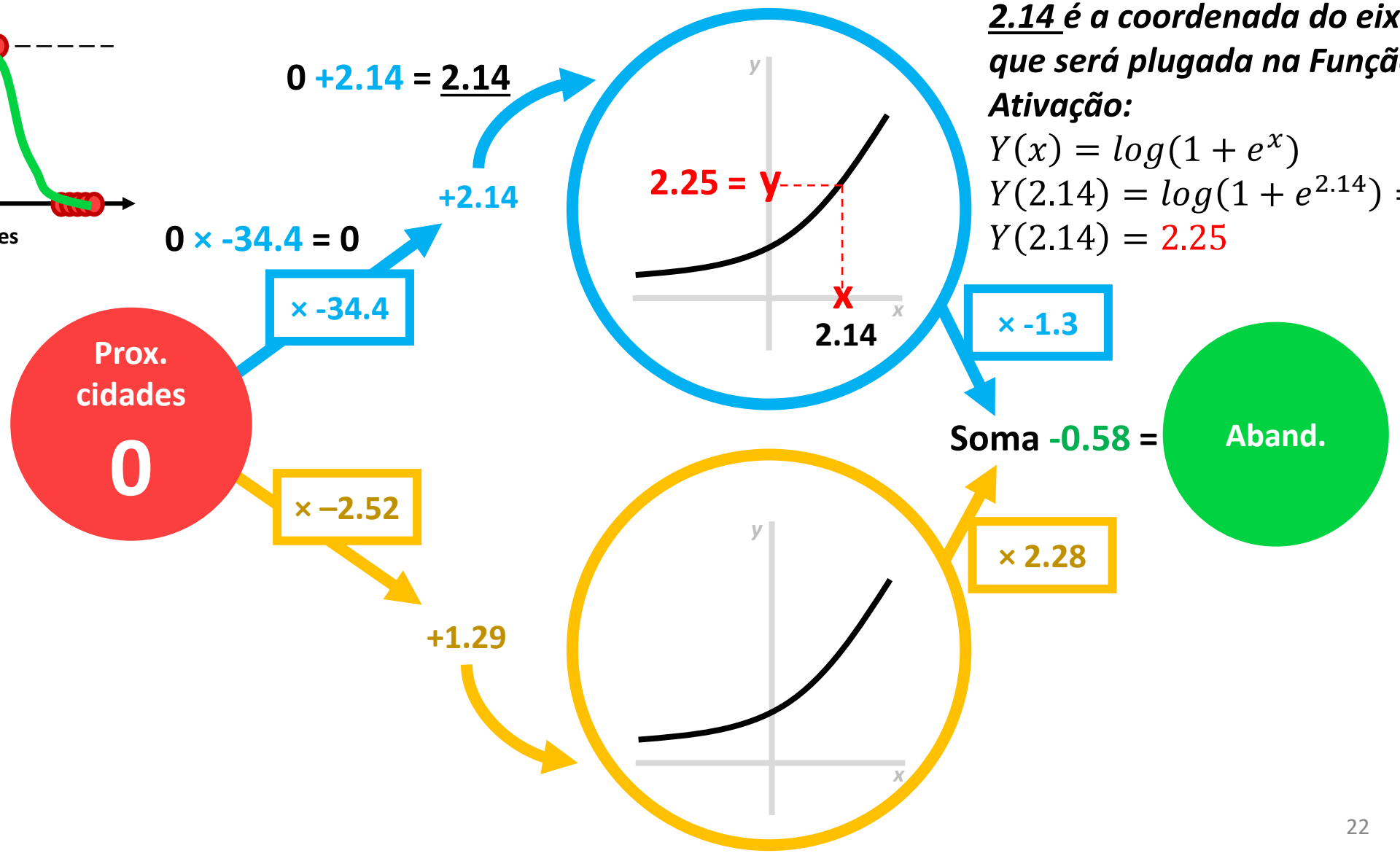
Como funciona uma Rede Neural Artificial (RNA)?





Como funciona uma Rede Neural Artificial (RNA)?





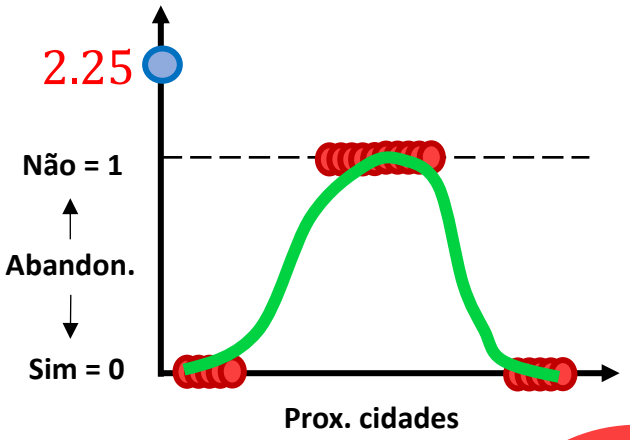
***2.14** é a coordenada do eixo x que será plugada na Função de Ativação:*

$$Y(x) = log(1 + e^{x})$$

$$Y(2.14) = log(1 + e^{2.14}) =$$

$$Y(2.14) = 2.25$$

Como funciona uma Rede Neural Artificial (RNA)?



Prox. cidades

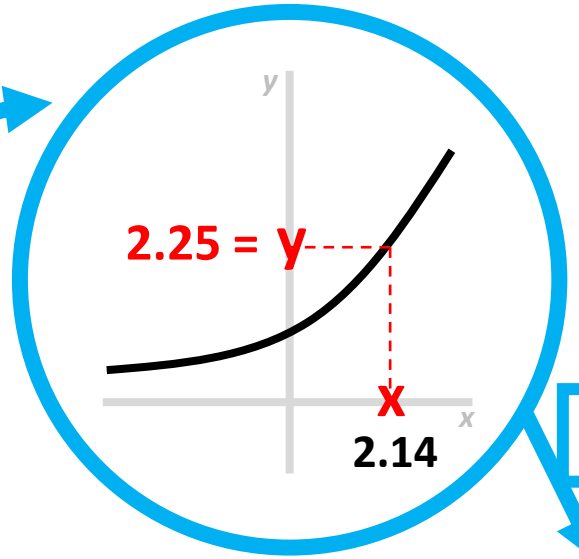
0

× -34.4

0 × -34.4 = 0

+2.14

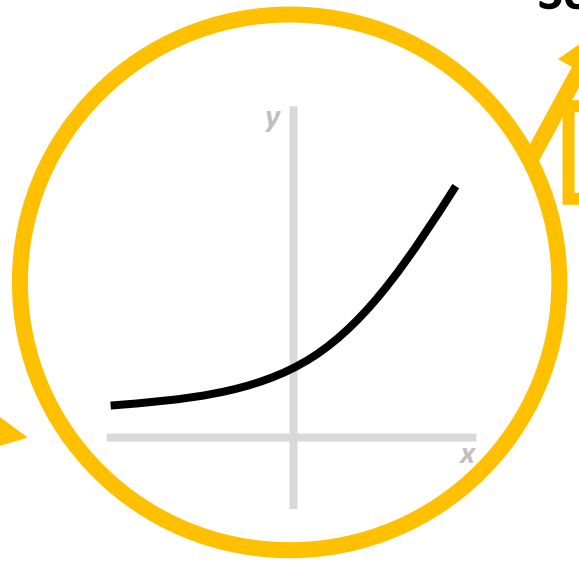
0 +2.14 = 2.14



× -1.3

× –2.52

+1.29



× 2.28

Soma -0.58 =

Aband.

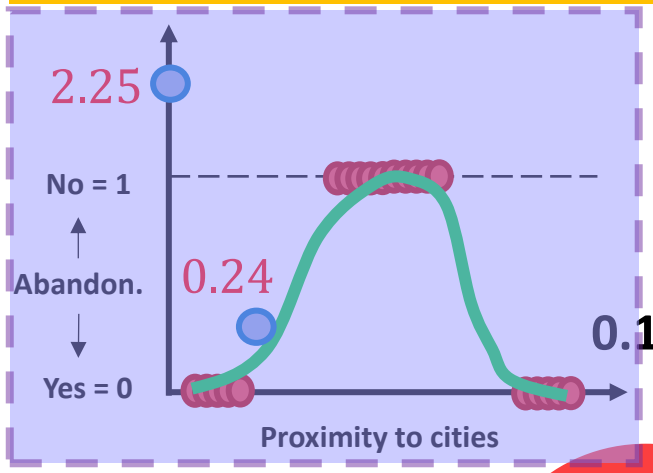
***2.14** é a coordenada do eixo x que será plugada na Função de Ativação:*

$$Y(x) = log(1 + e^{x})$$

$$Y(2.14) = log(1 + e^{2.14}) =$$

$$Y(2.14) = 2.25$$

Como funciona uma Rede Neural Artificial (RNA)?



Prox. cidades

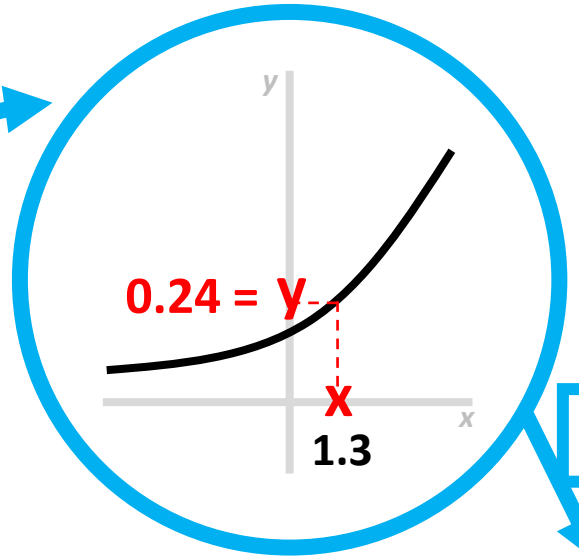
0.1

0.1 × -34.4 = -3.44

× -34.4

+2.14

-3.44 +2.14 = 1.3



***1.3** é a coordenada do eixo x que será plugada na Função de Ativação:*

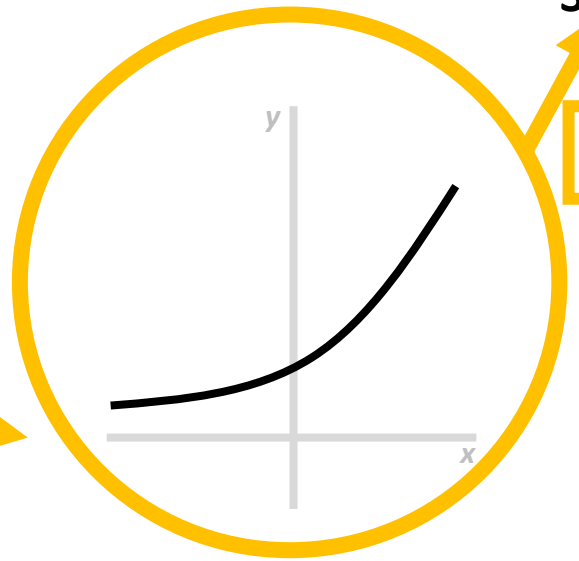
$$Y(x) = log(1 + e^{x})$$

$$Y(1.3) = log(1 + e^{1.3}) =$$

$$Y(1.3) = 0.24$$

× –2.52

+1.29



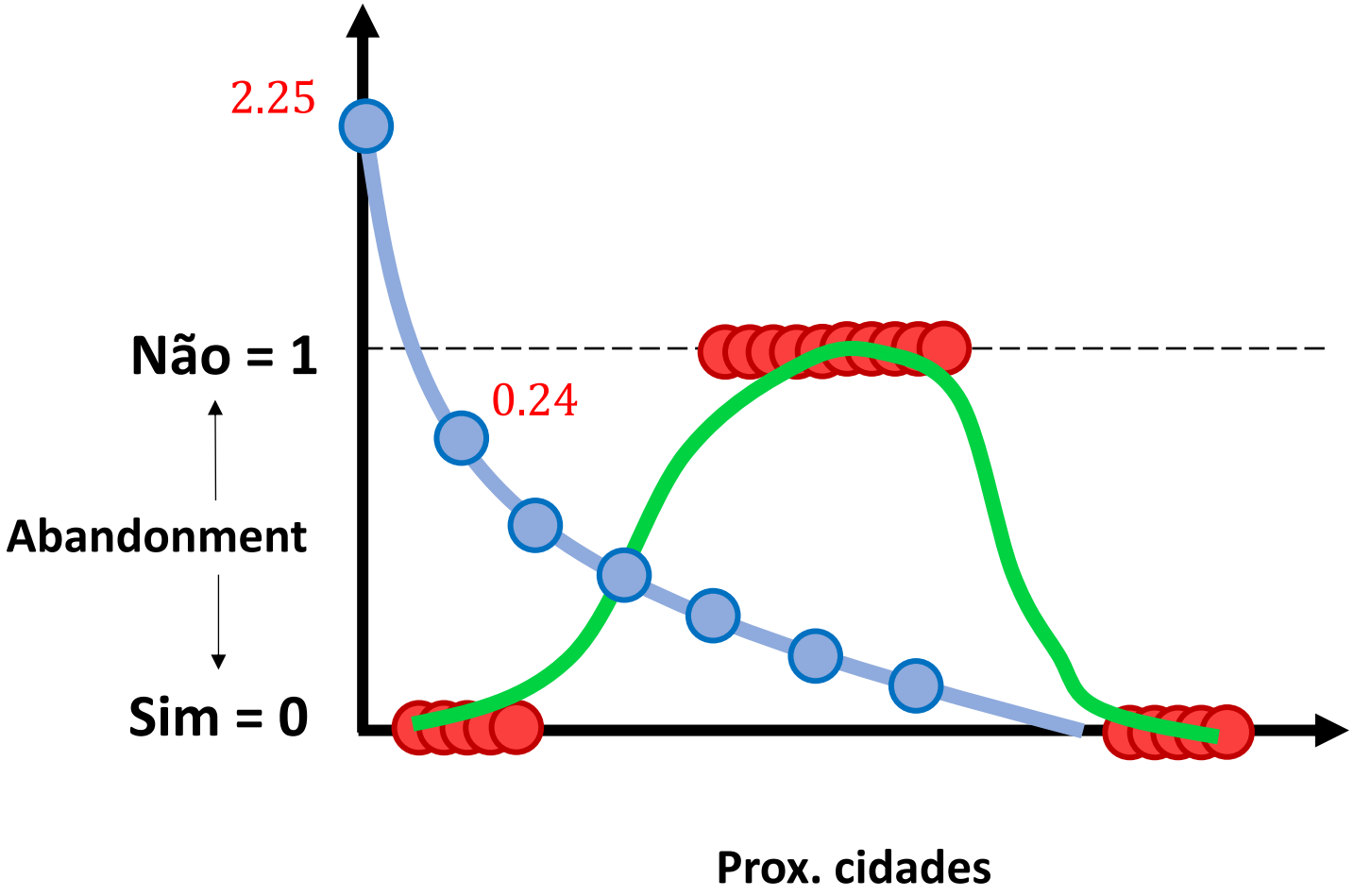
× -1.3

× 2.28

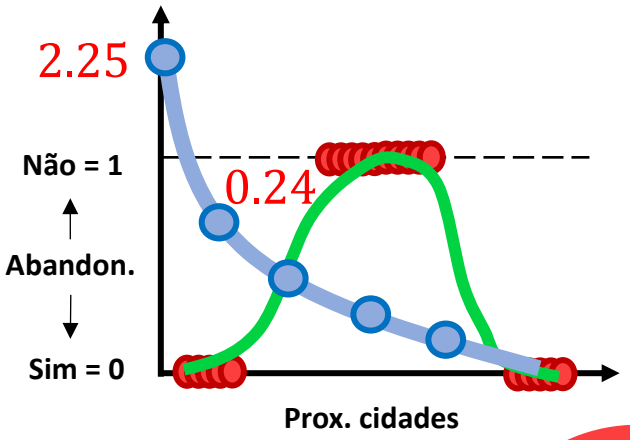
Sum +2.58 =

Aband.

Como funciona uma Rede Neural Artificial (RNA)?



Como funciona uma Rede Neural Artificial (RNA)?



Prox. cidades

× -34.4

+2.14

× –2.52

+1.29

y
x

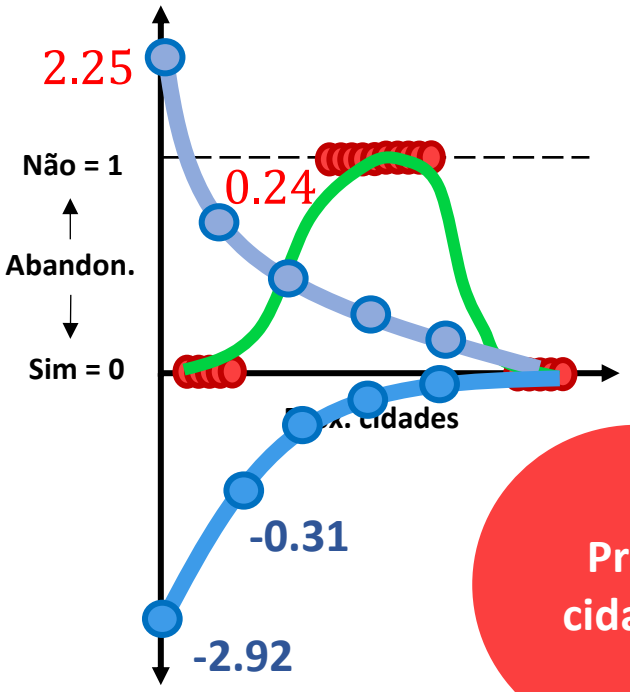
× -1.3

Soma +2.58 =

Aband.

× 2.28

Como funciona uma Rede Neural Artificial (RNA)?



Prox. cidades

× -34.4

+2.14

× –2.52

+1.29

y
x

× -1.3

× 2.28

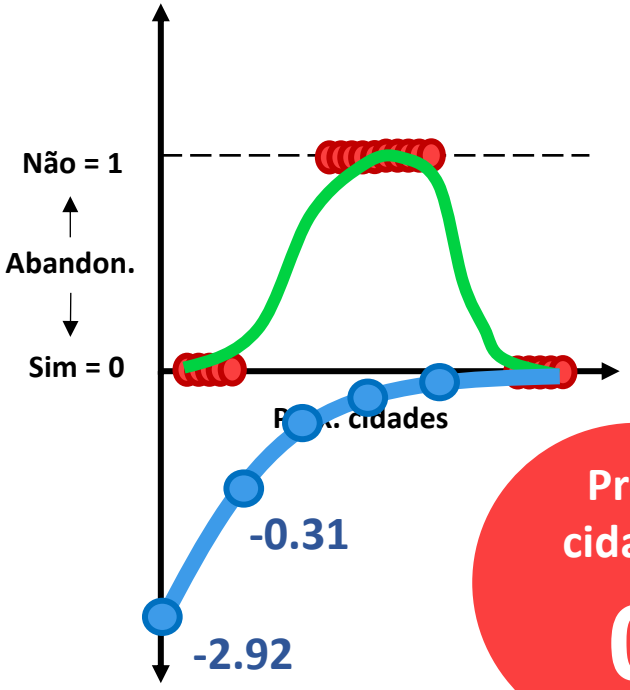
2.25 × -1.3 = -2.92

0.24 × -1.3 = -0.31

...

Soma +2.58 = Aband.

Como funciona uma Rede Neural Artificial (RNA)?



Prox. cidades

0

× -34.4

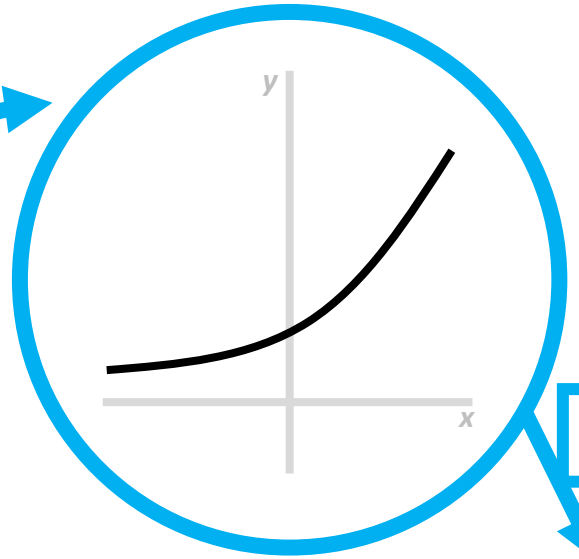
+2.14

× –2.52

0 × –2.52 = 0

+1.29

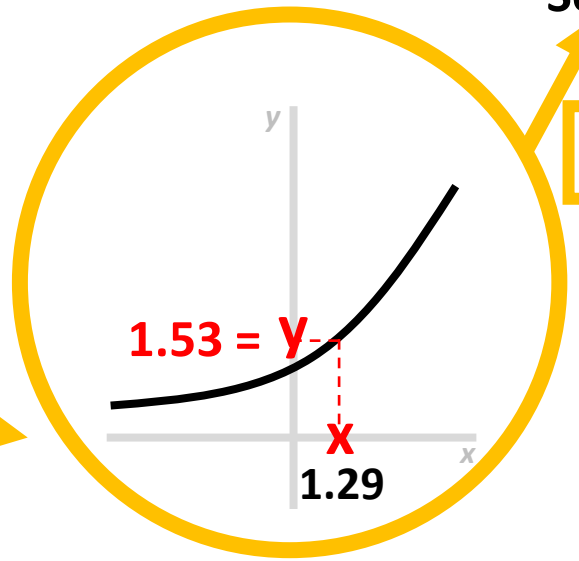
0 +1.29 = 1.29



× -1.3

Soma +2.58 =

Aband.



× 2.28

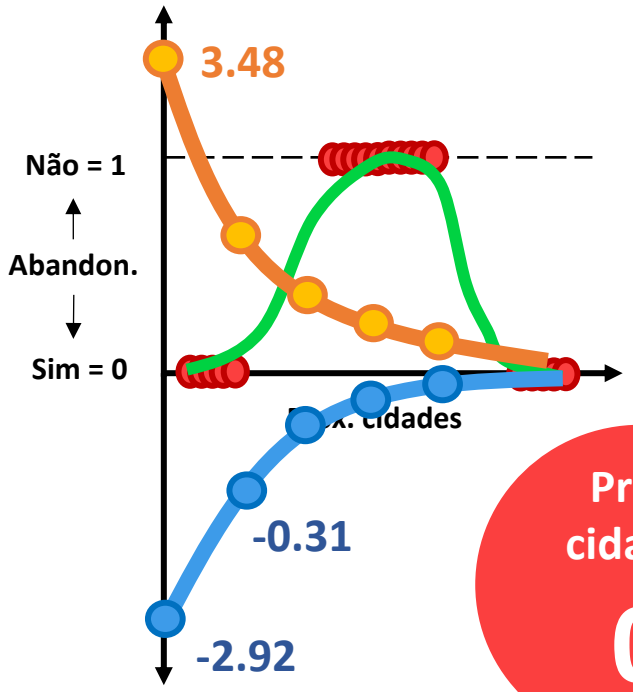
***1.29** é a coordenada do eixo x que será plugada na Função de Ativação:*

$Y(x) = log(1 + e^x)$

$Y(1.29) = log(1 + e^{1.29}) =$

$Y(1.29) = 1.53$

Como funciona uma Rede Neural Artificial (RNA)?



Prox. cidades

0

× -34.4

+2.14

× –2.52

0 × –2.52 = 0

+1.29

0 +1.29 = 1.29

1.53 = y

x

1.29

× -1.3

Soma +2.58 =

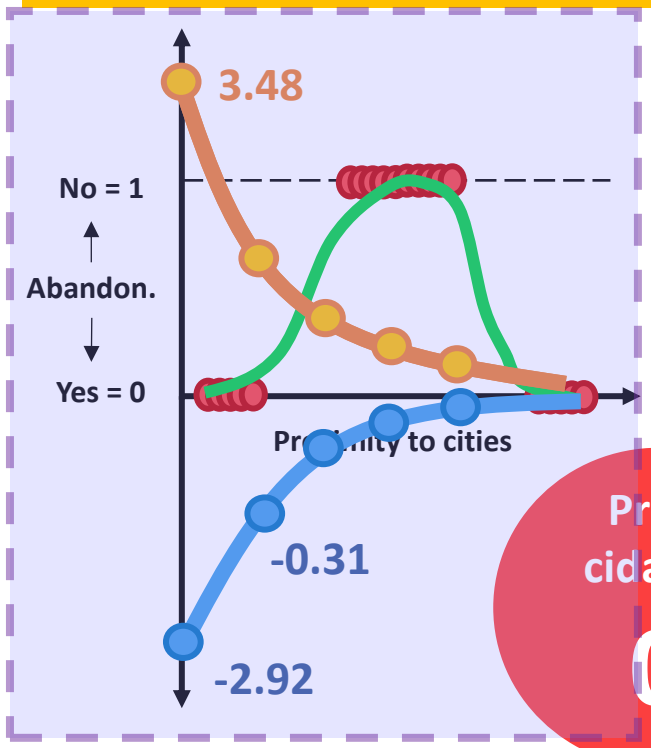
Aband.

× 2.28

1.53 × 2.28 = 3.48

...

Como funciona uma Rede Neural Artificial (RNA)?



Prox. cidades

0

× -34.4

+2.14

× –2.52

0 × –2.52 = 0

+1.29

0 +1.29 = 1.29

1.53 = y

x

1.29

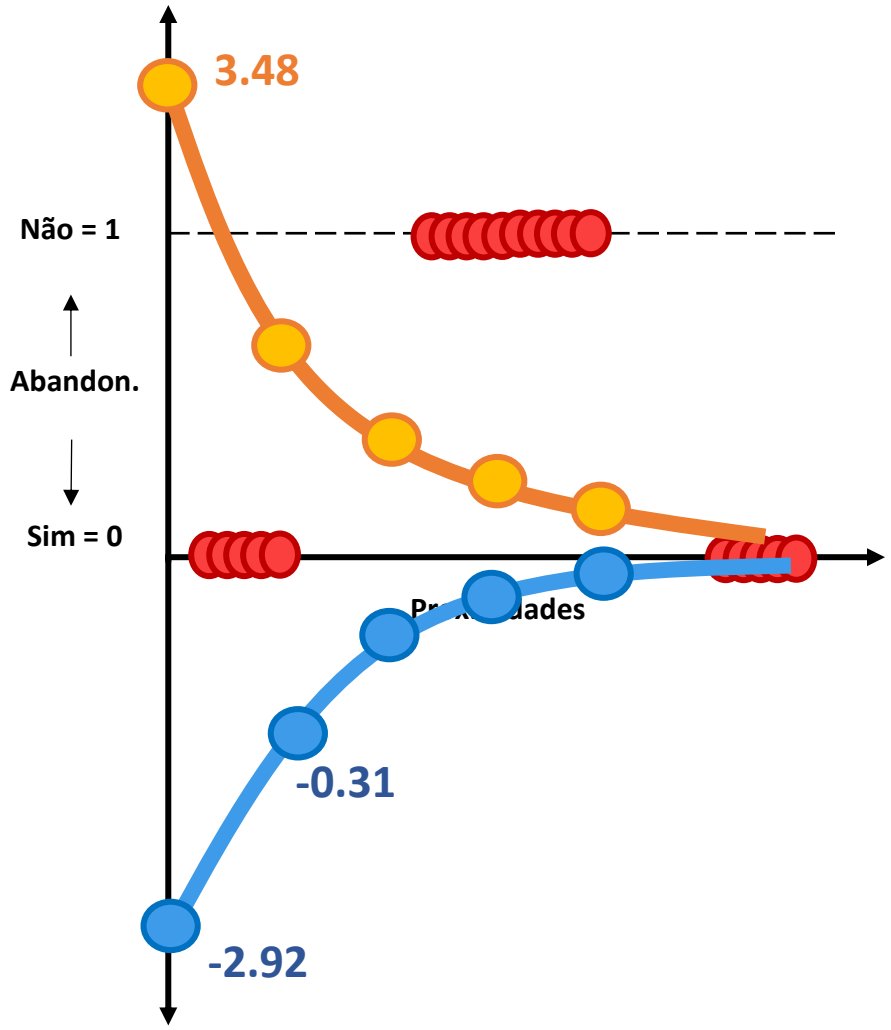
× -1.3

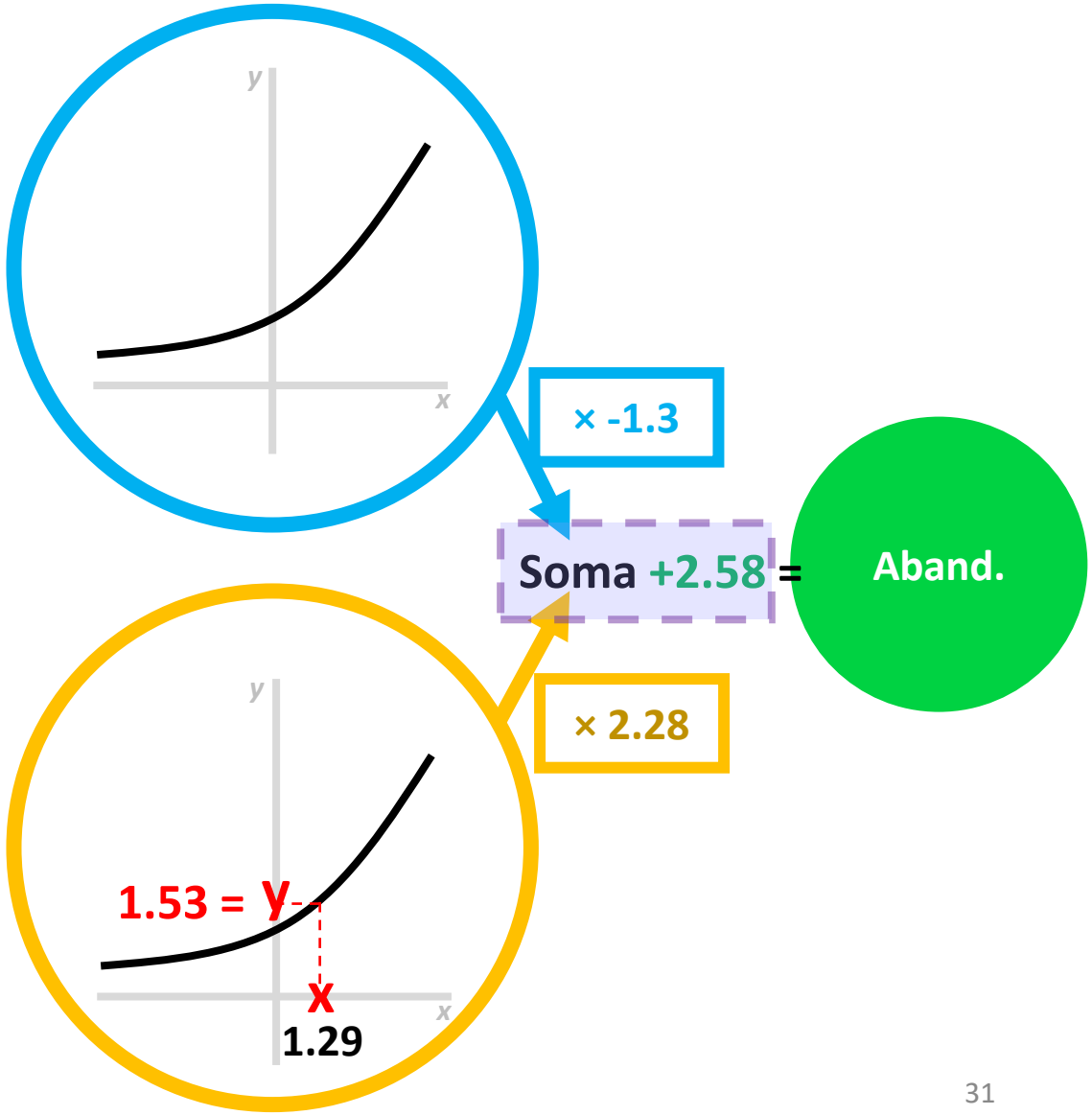
× 2.28

Soma +2.58 =

Aband.

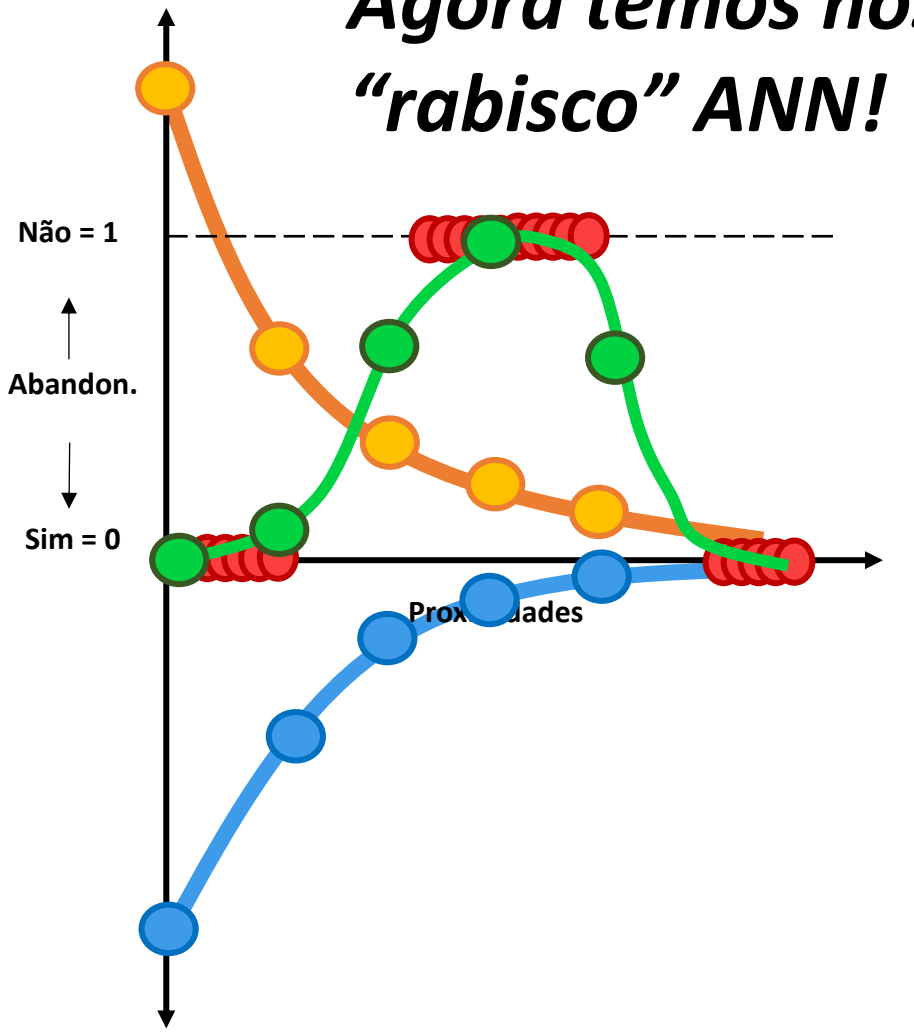
Como funciona uma Rede Neural Artificial (RNA)?

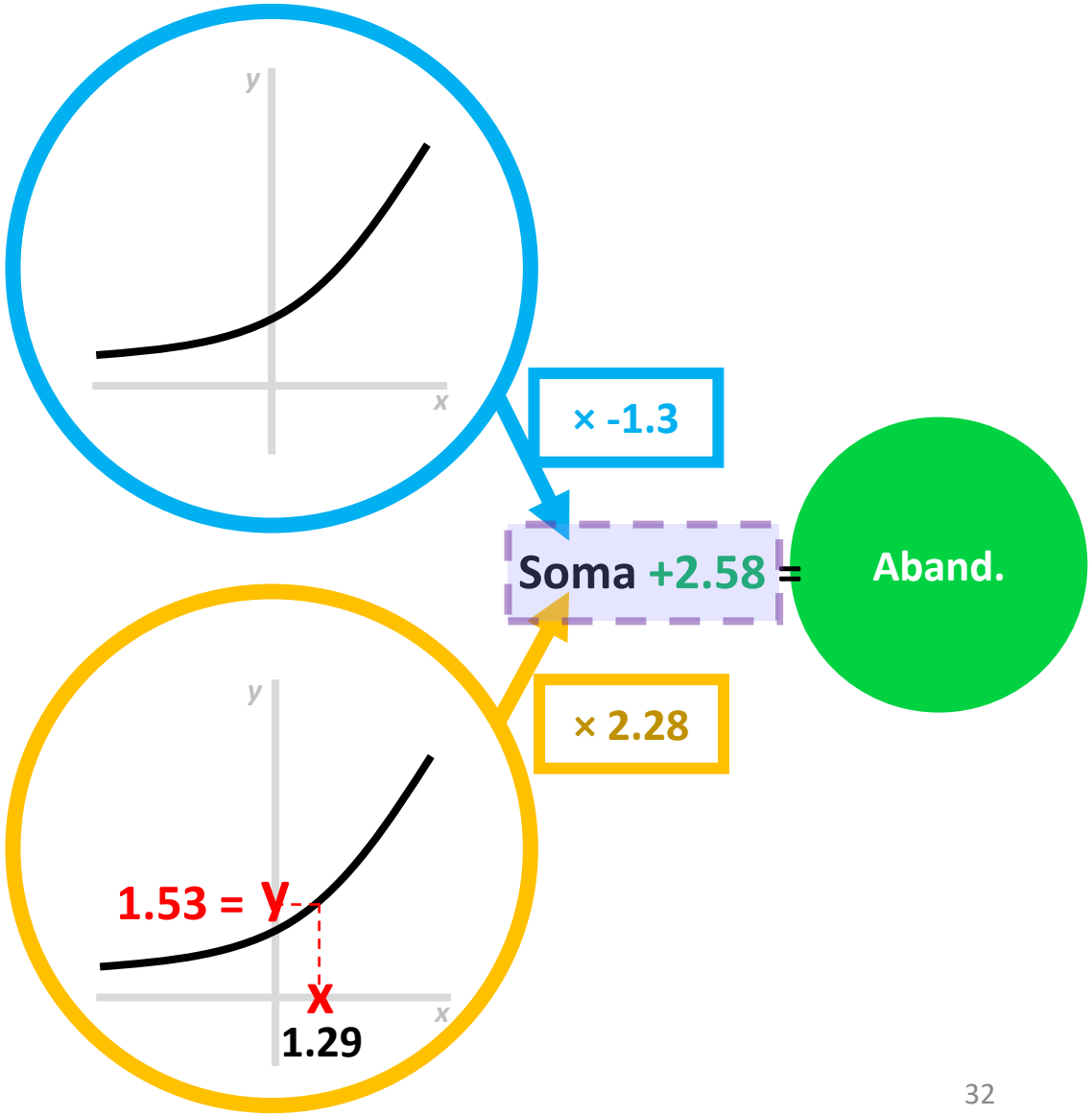




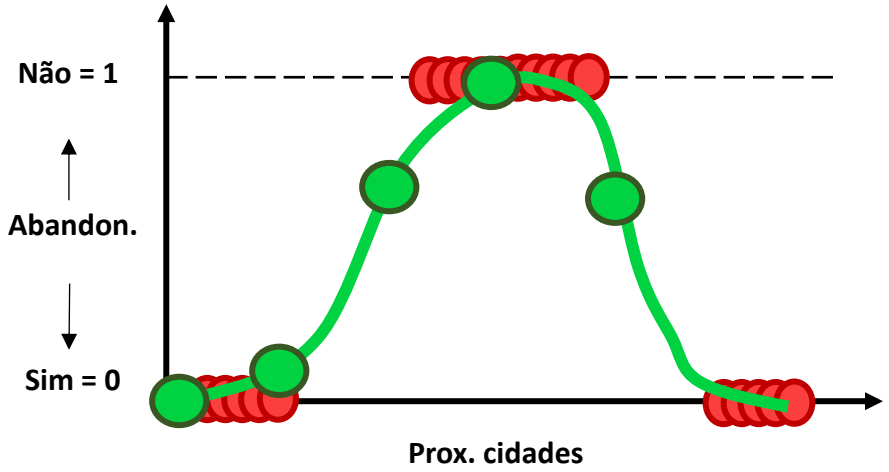
Como funciona uma Rede Neural Artificial (RNA)?

Agora temos nosso "rabisco" ANN!



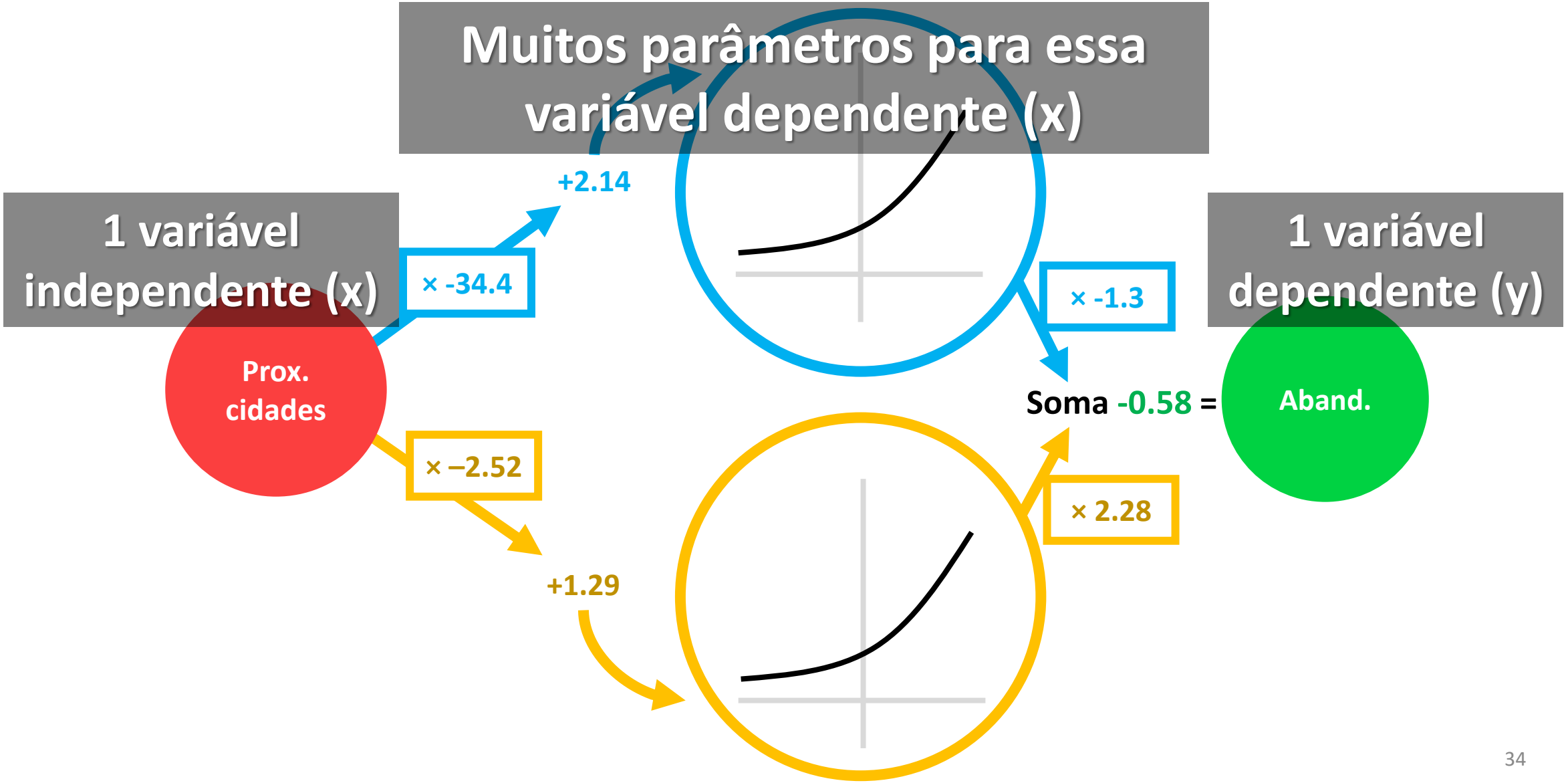


Como funciona uma Rede Neural Artificial (RNA)?



Agora podemos inserir valores em nossa RNA e obter valores (por meio do "rabisco") da probabilidade de abandono de terras com base na proximidade das cidades

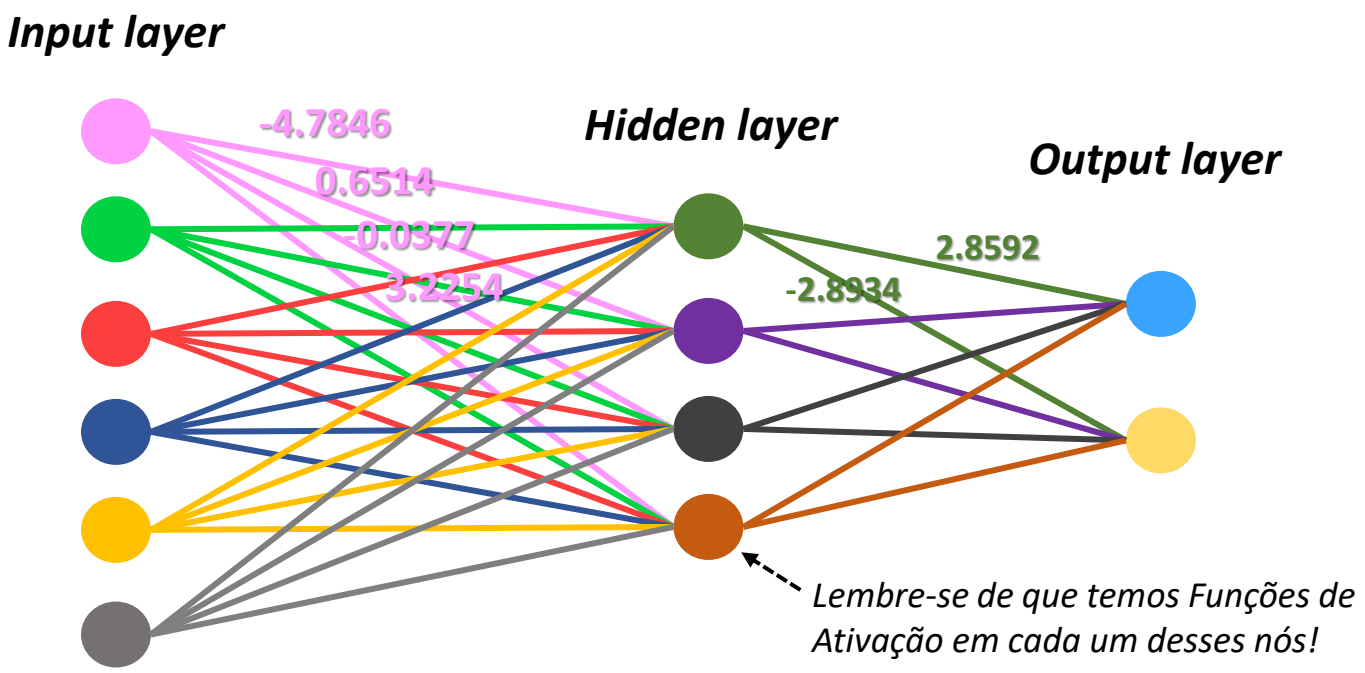
Todo esse trabalho para uma RNA muito simples ...



RNA são geralmente muito mais complexas...

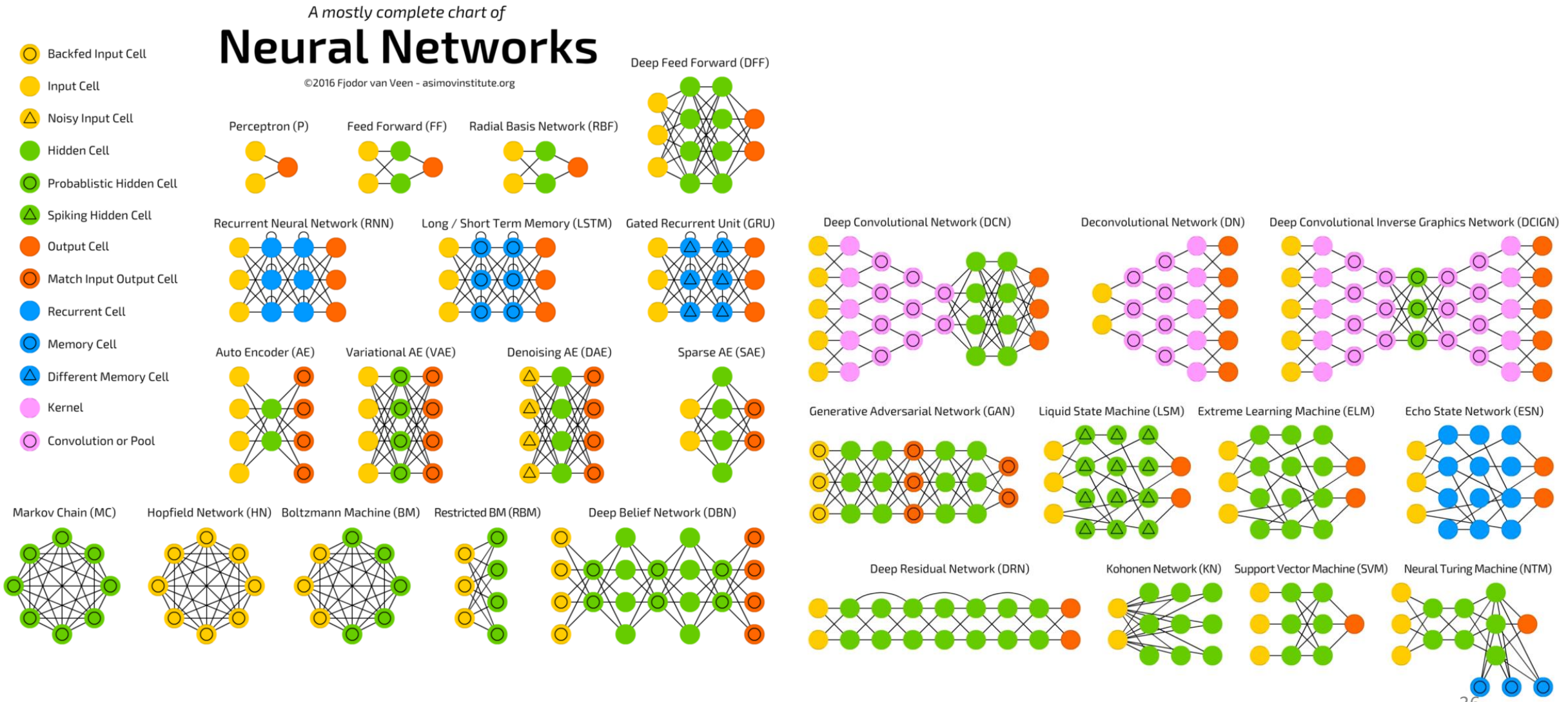
	Hidden Node 1	Hidden Node 2	Hidden Node 3	Hidden Node 4
Input Node/Variável 1	-4.7846	0.6514	-0.0377	3.2254
Input Node/Variável 2	0.4254	0.1605	0.0891	-0.2653
Input Node/Variável 3	0.5620	0.2345	0.0892	-0.4920
Input Node/Variável 4	0.5114	0.2038	0.1367	-0.3558
Input Node/Variável 5	0.3542	0.3040	0.1179	-0.1726
Input Node/Variável 6	-2.3447	0.6591	0.1593	2.2832

Muitos parâmetros... Impossível de interpretar! "Caixa preta"

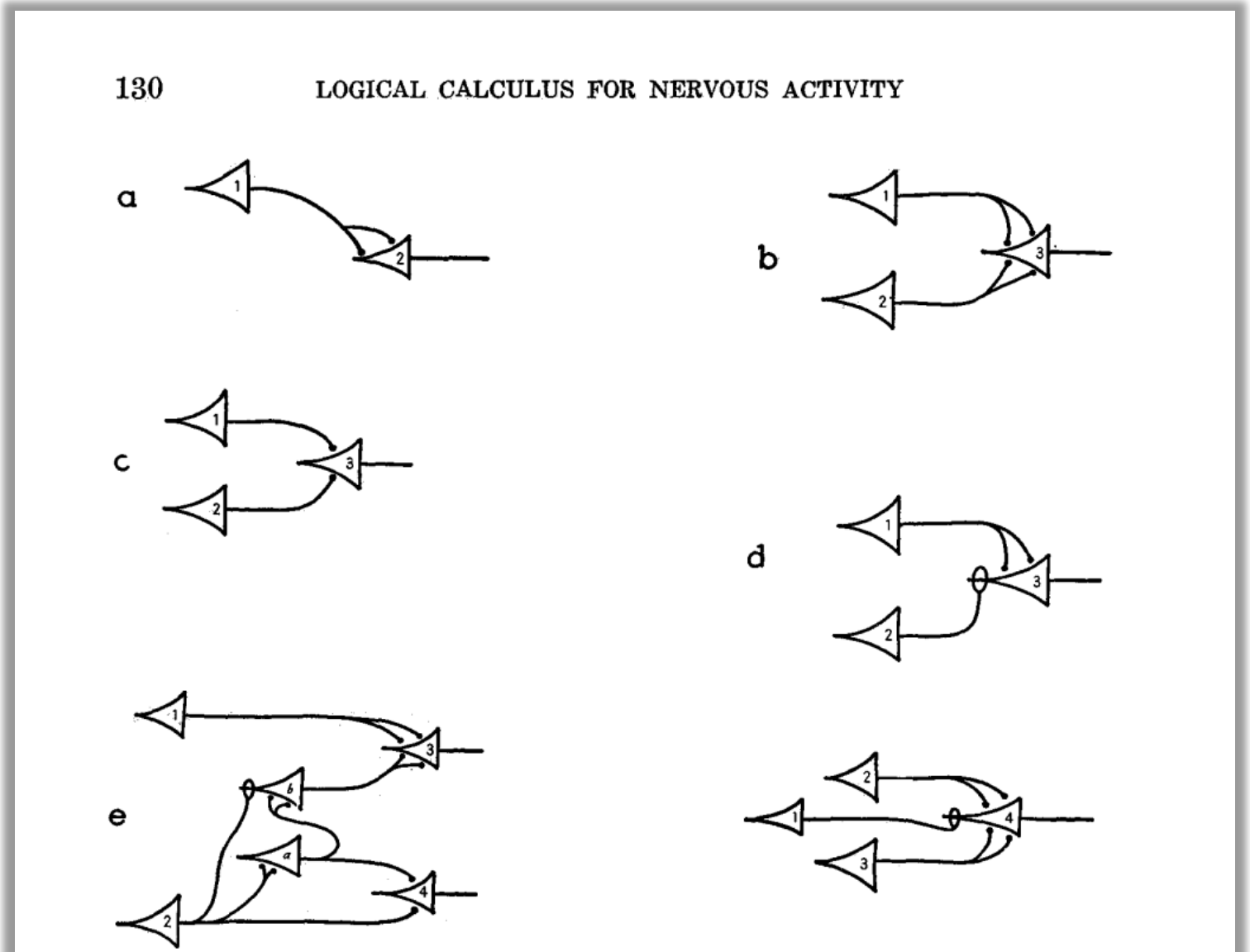


	Output Neuron 1	Output Neuron 2
Hidden Node 1	2.8592	-2.8934
Hidden Node 2	-1.2837	1.1892
Hidden Node 3	-0.4875	0.6556
Hidden Node 4	-3.1082	3.0860

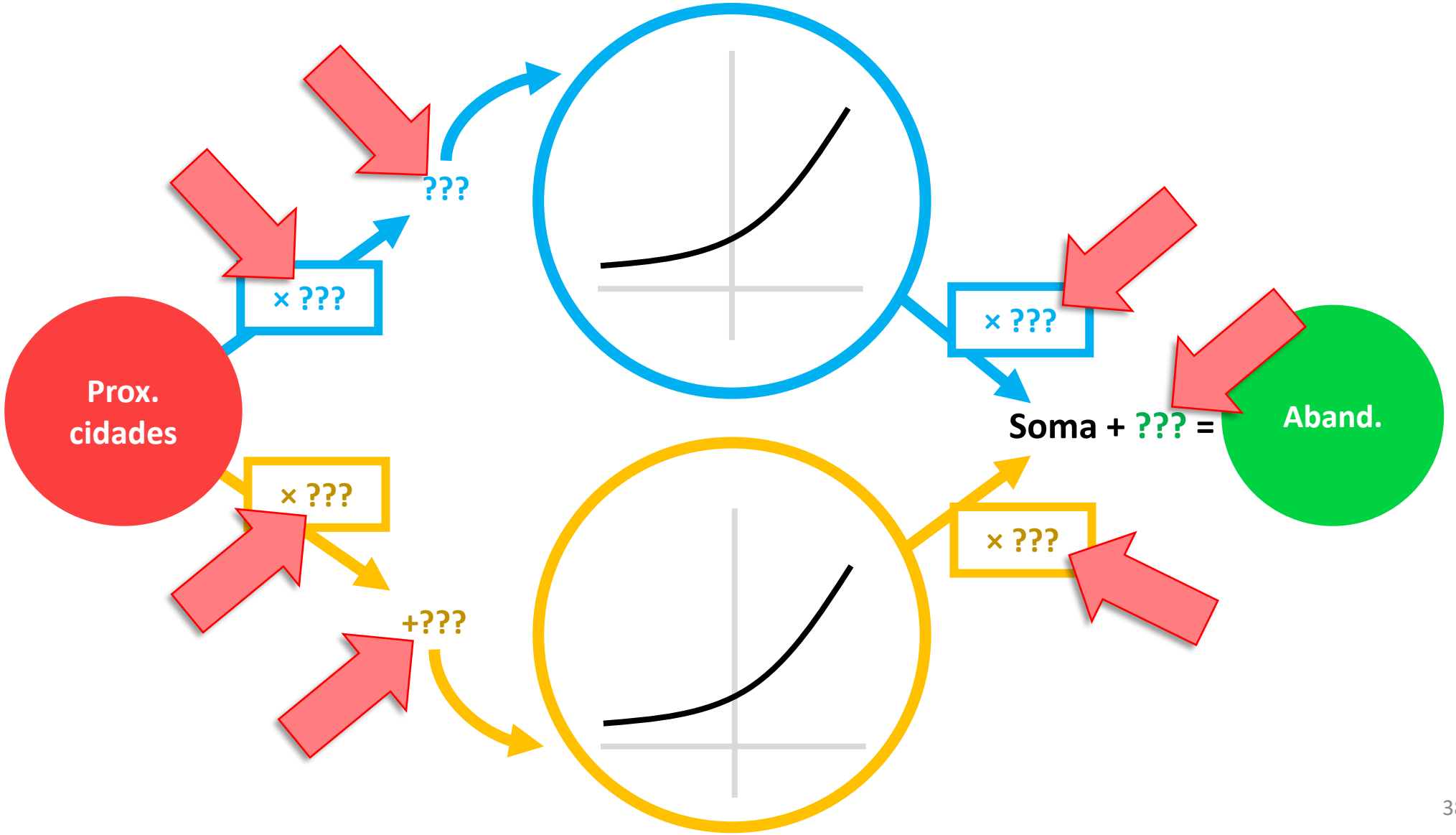
RNA são geralmente muito mais complexas...



RNA são geralmente muito mais complexas…



Como estimar parâmetros em RNA?



Aula: Duas Partes

Parte 1. O que são e como funcionam as RNA?

Parte 2. Como estimar parâmetros em RNA?

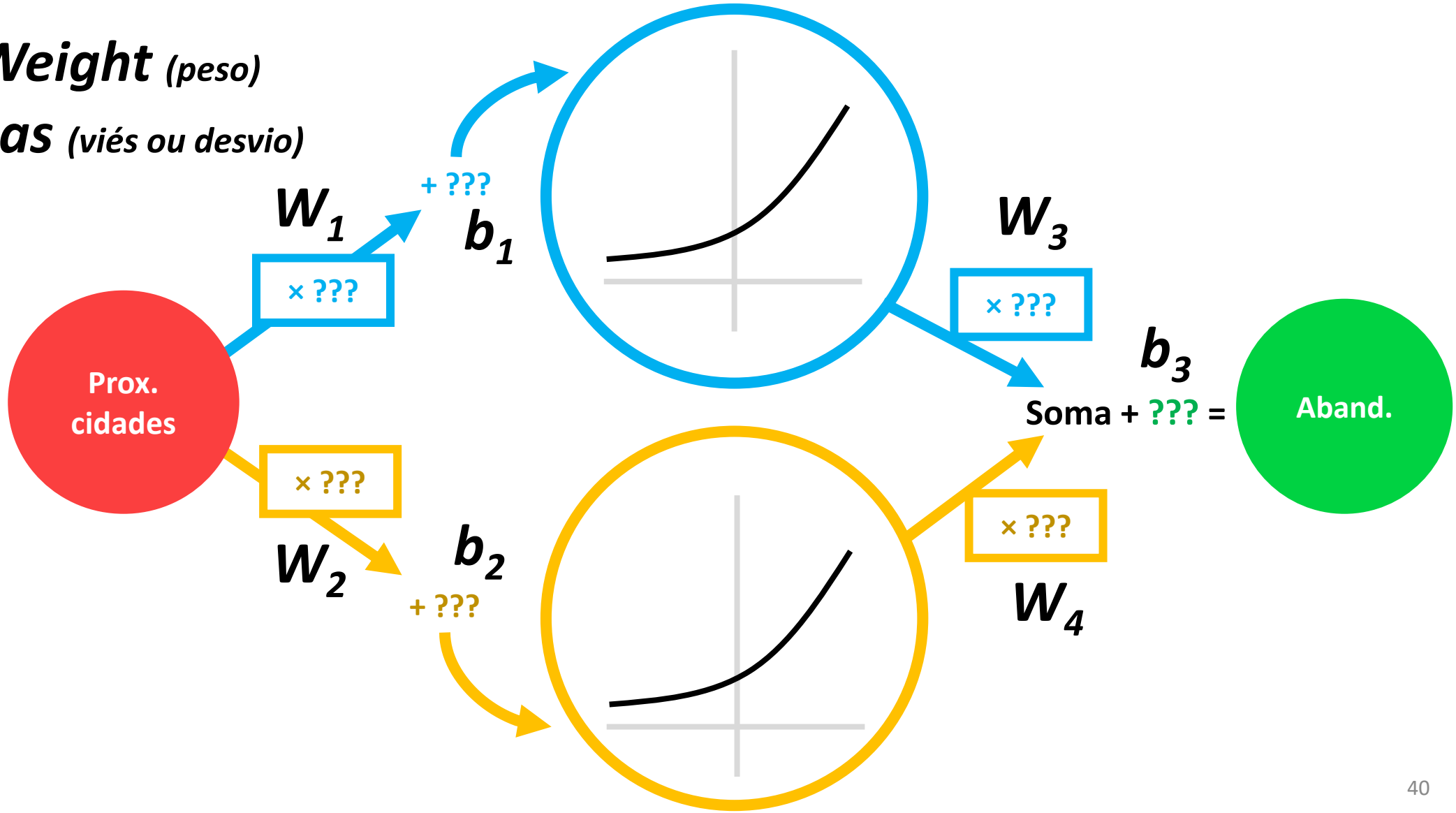
RETROPROPAGAÇÃO

(BACKPROPAGATION)

Como estimar parâmetros em uma RNA?

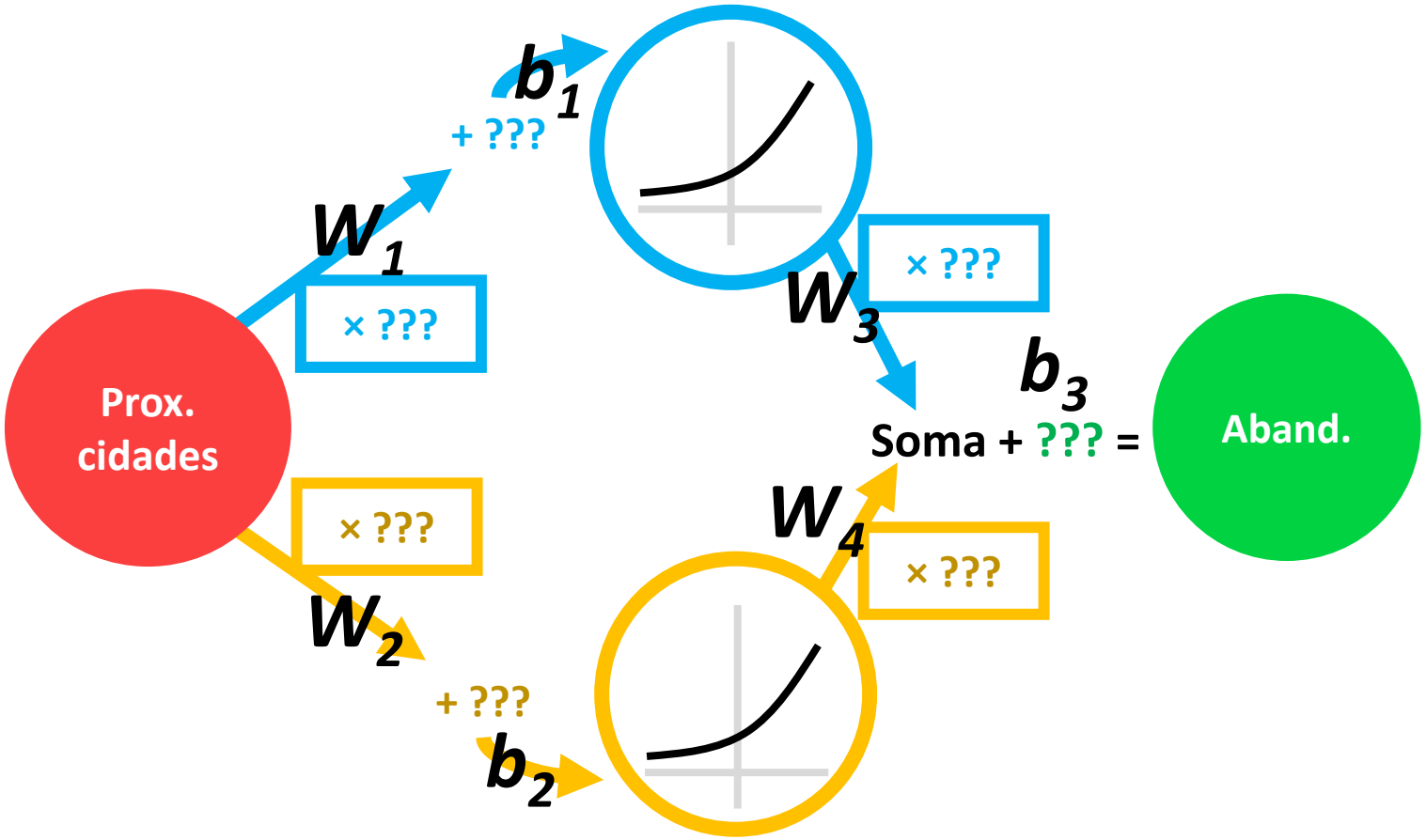
***W = Weight** (peso)*

***b = Bias** (viés ou desvio)*



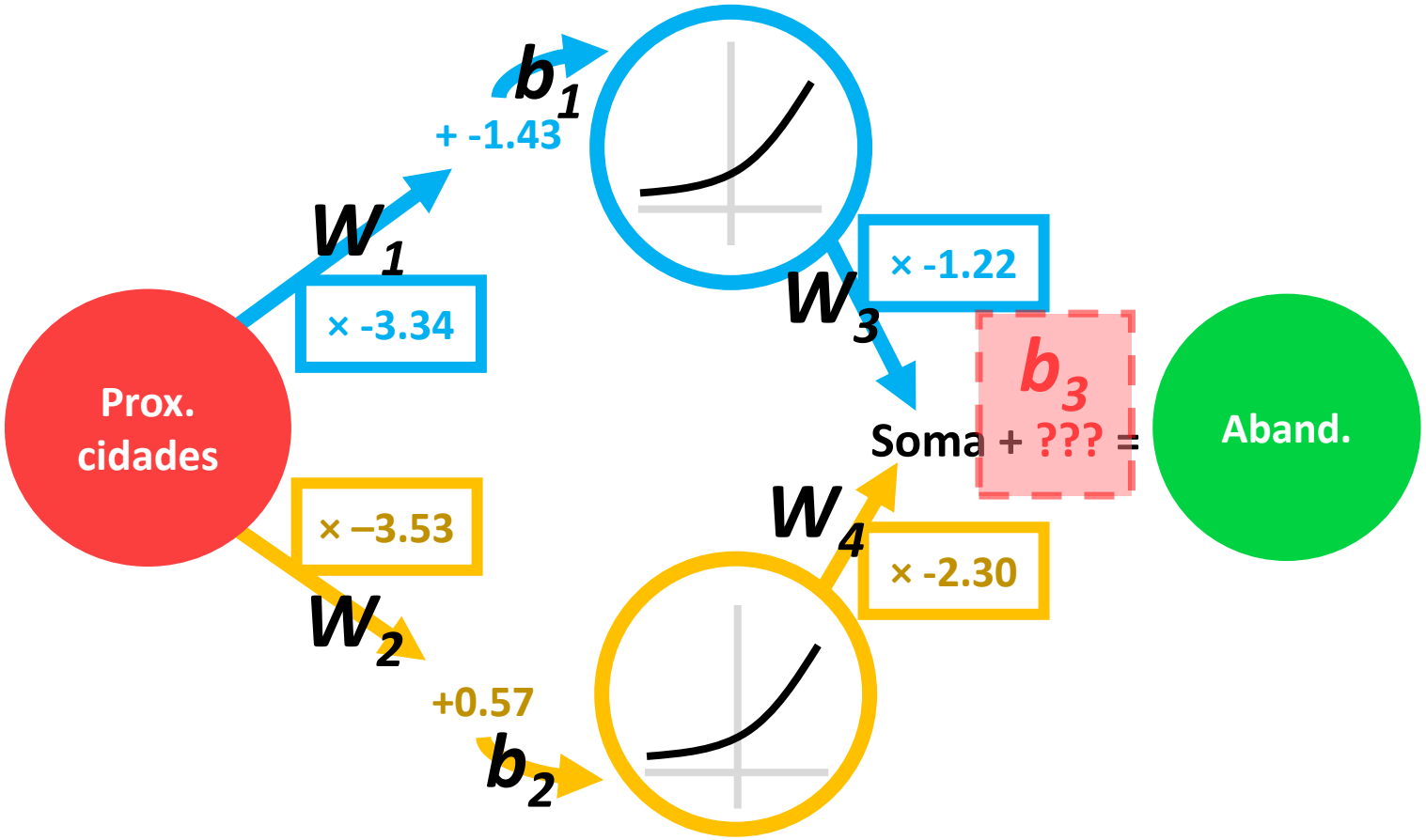
Como estimar parâmetros em uma RNA?

Vamos torná-la menor para economizar espaço...

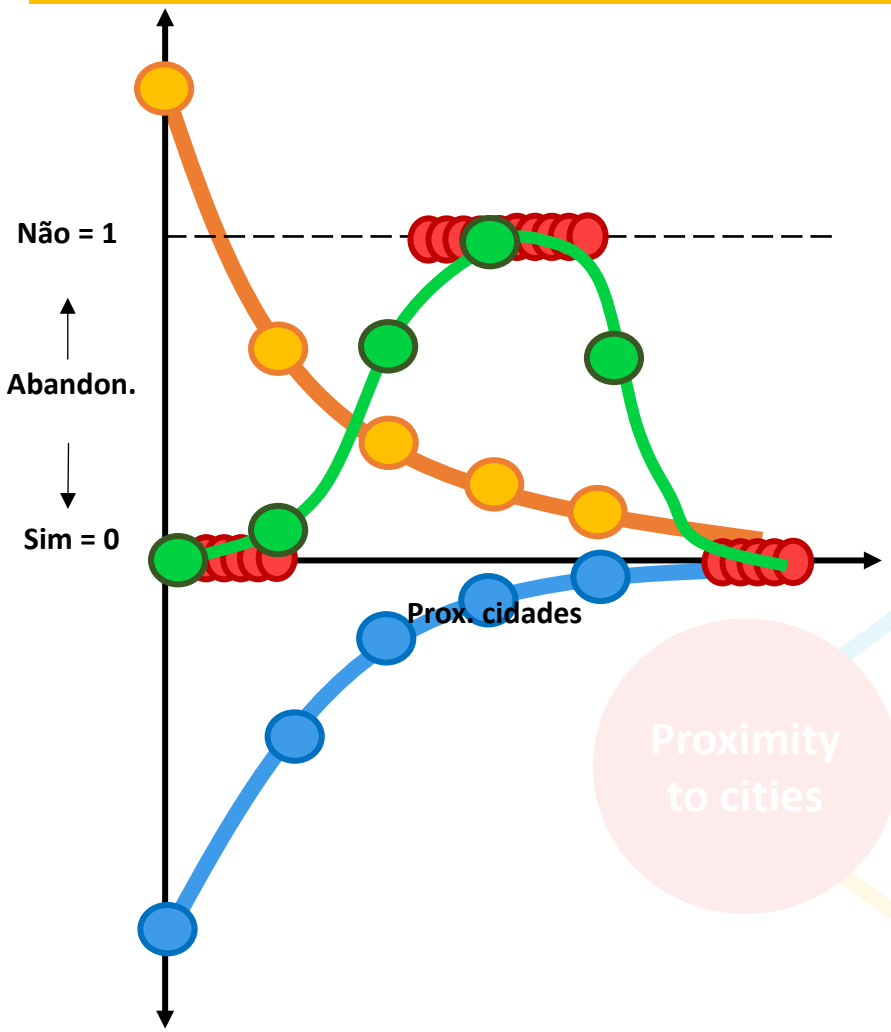


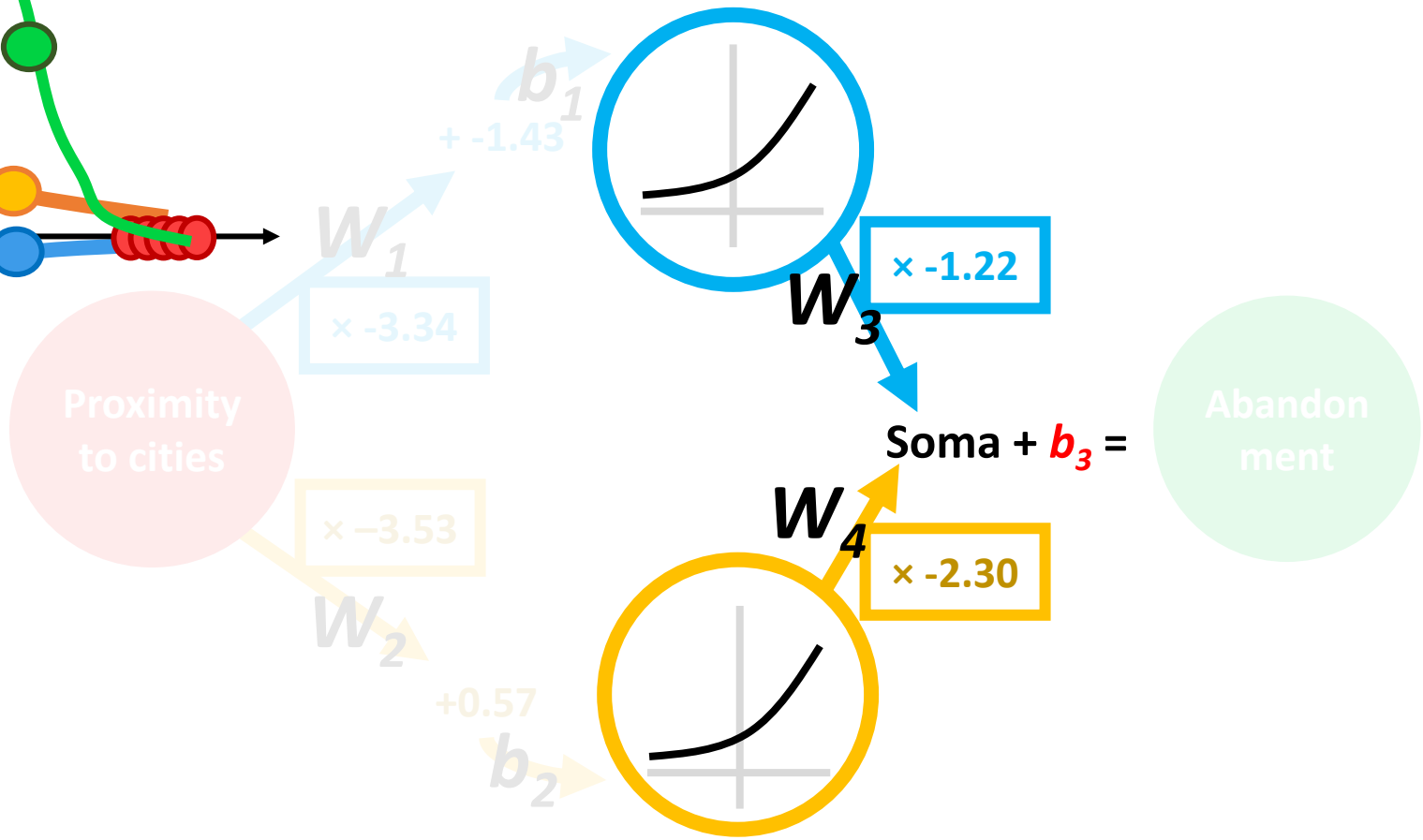
Como estimar parâmetros em uma RNA?

Por enquanto, vamos supor que já otimizamos todos os weights e biases, menos b3 ...

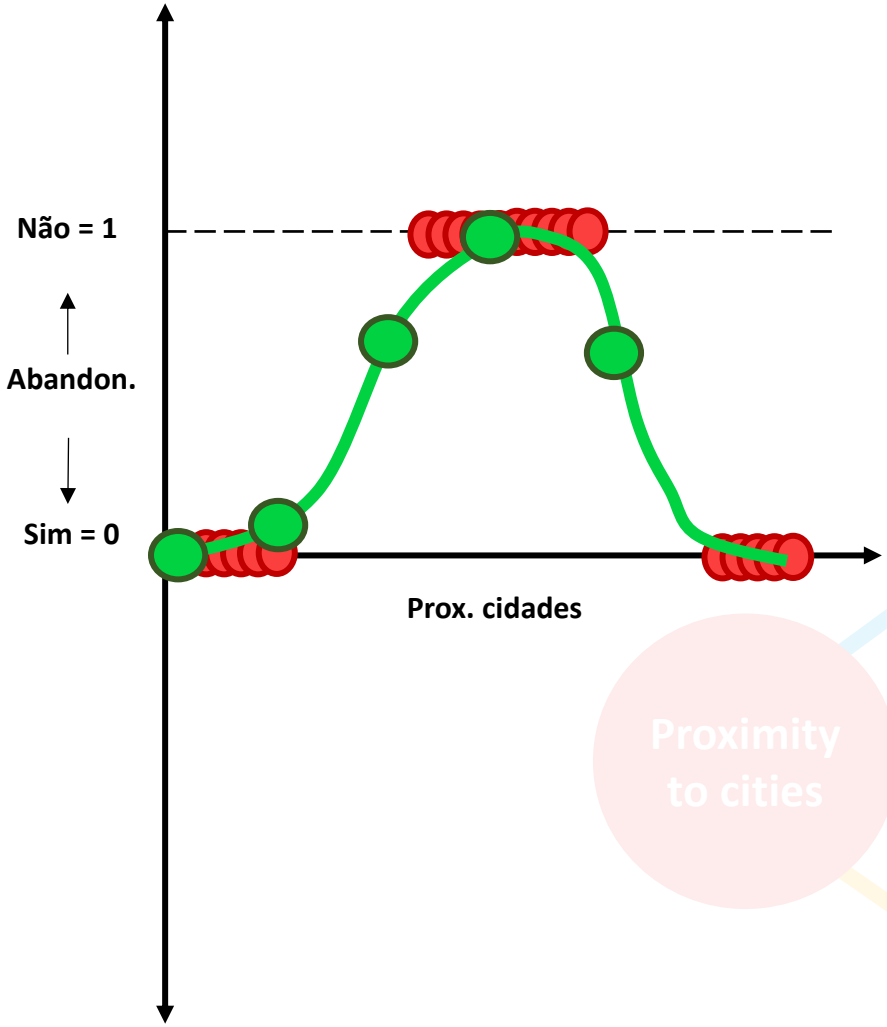


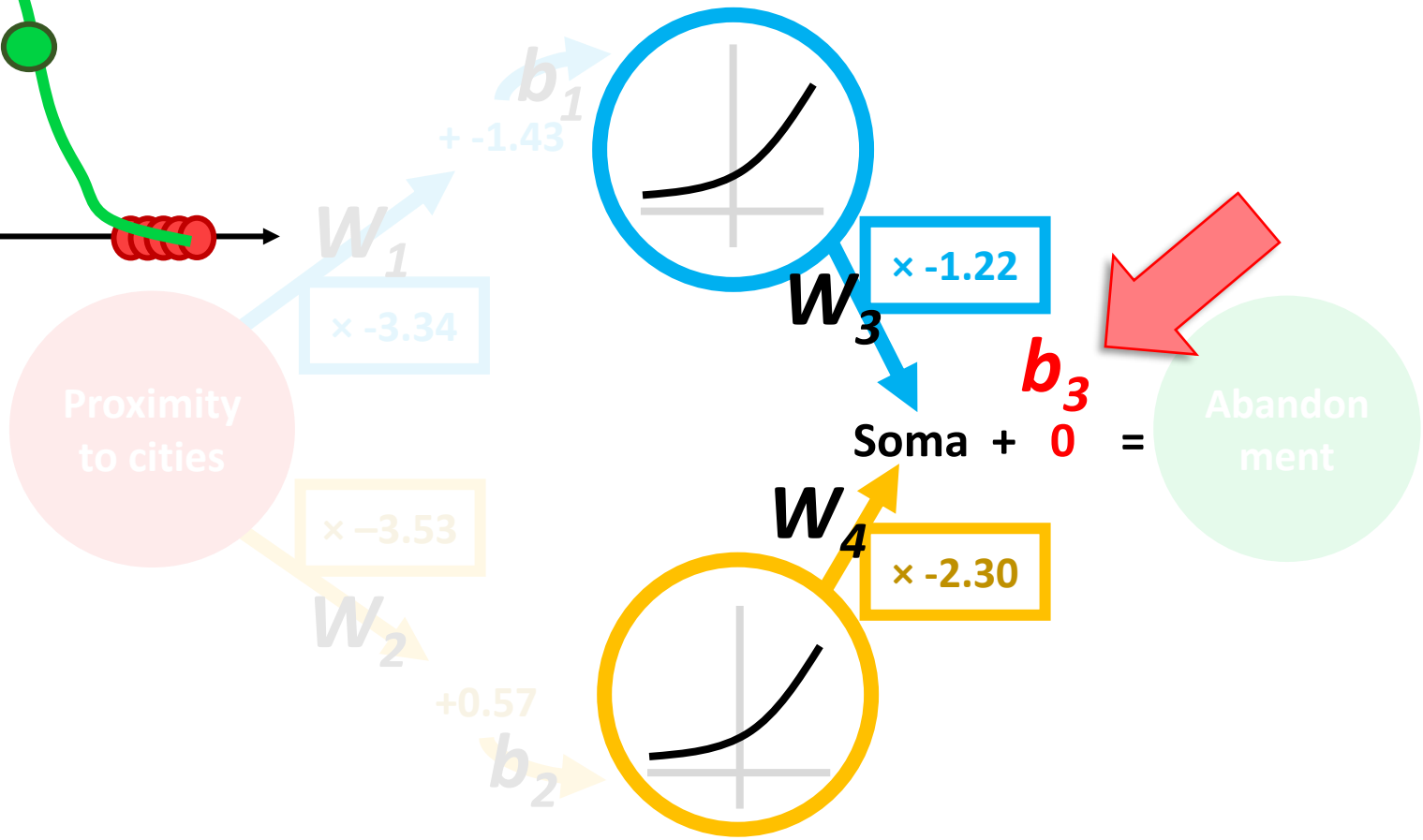
Como estimar parâmetros em uma RNA?



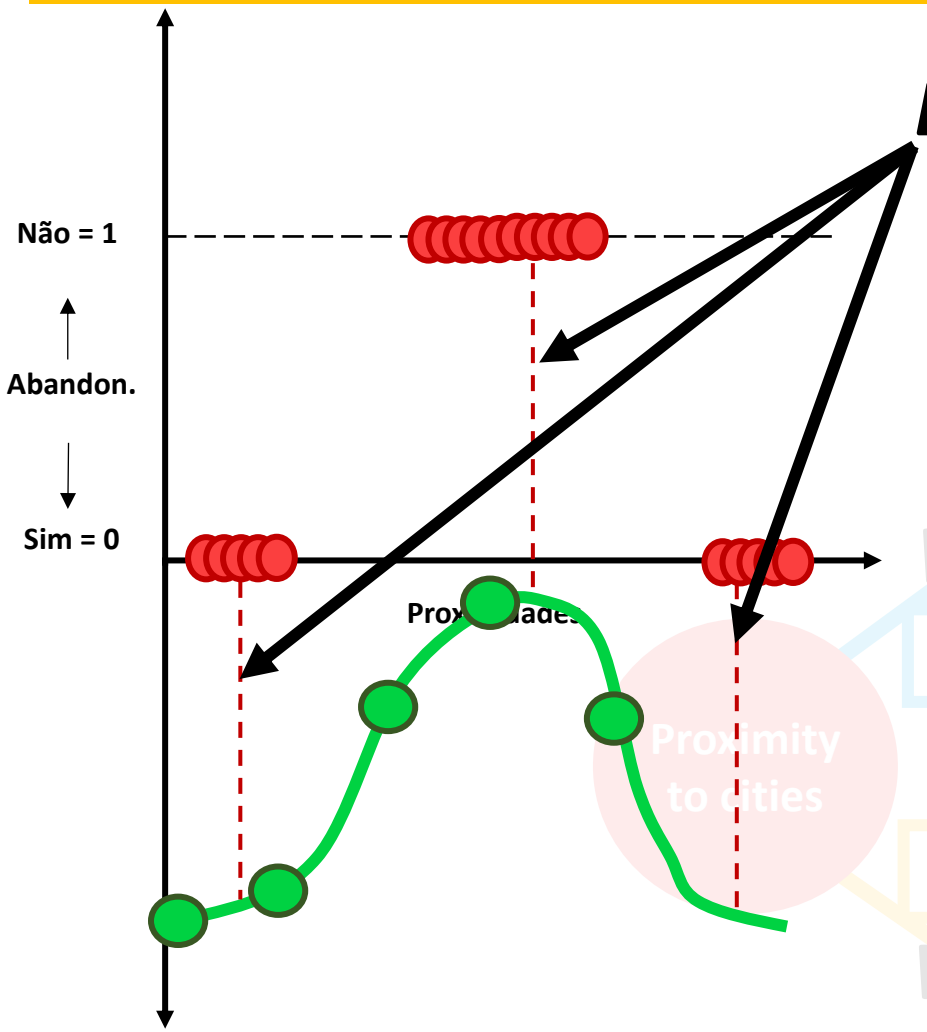


Como estimar parâmetros em uma ANN?





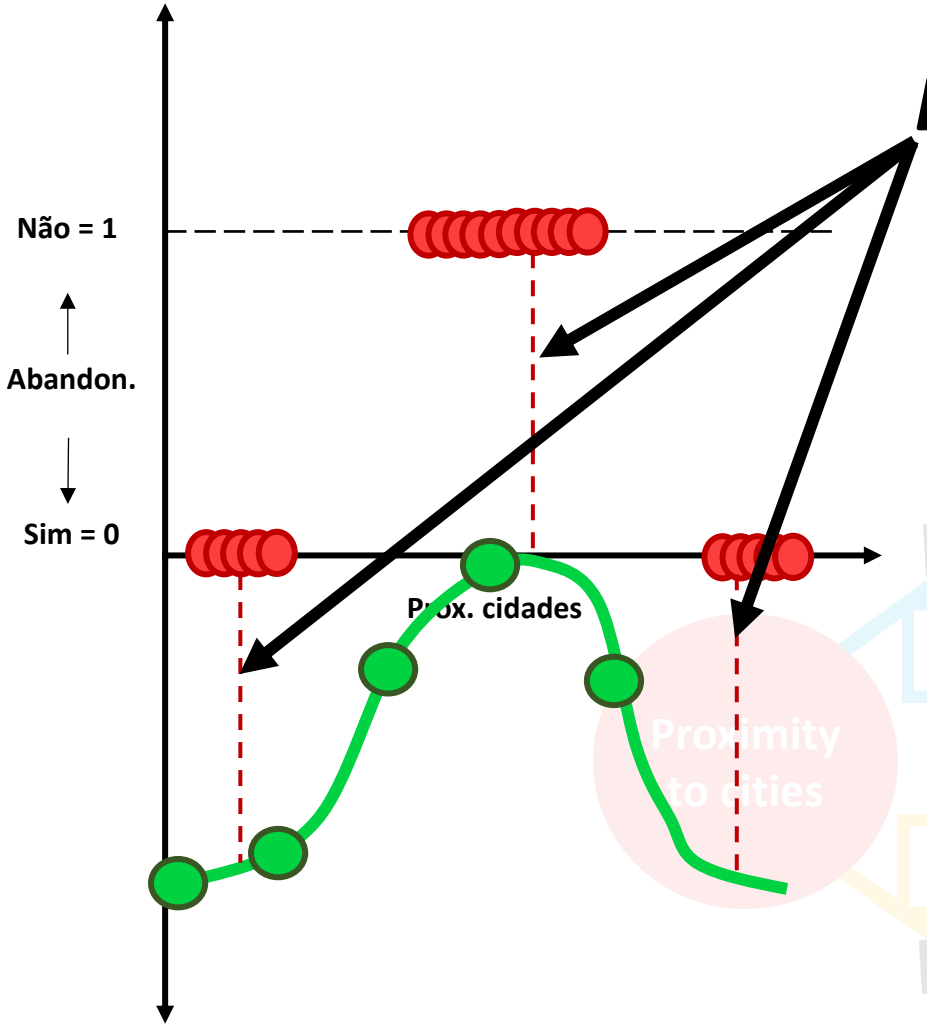
Como estimar parâmetros em uma RNA?



Essas diferenças são <u>resíduos</u>…

Neste caso, associados a $b_3 = 0$

Como estimar parâmetros em uma RNA?



Essas diferenças são <u>resíduos</u>...

E neste caso, associados a $b_3 = 0.1$

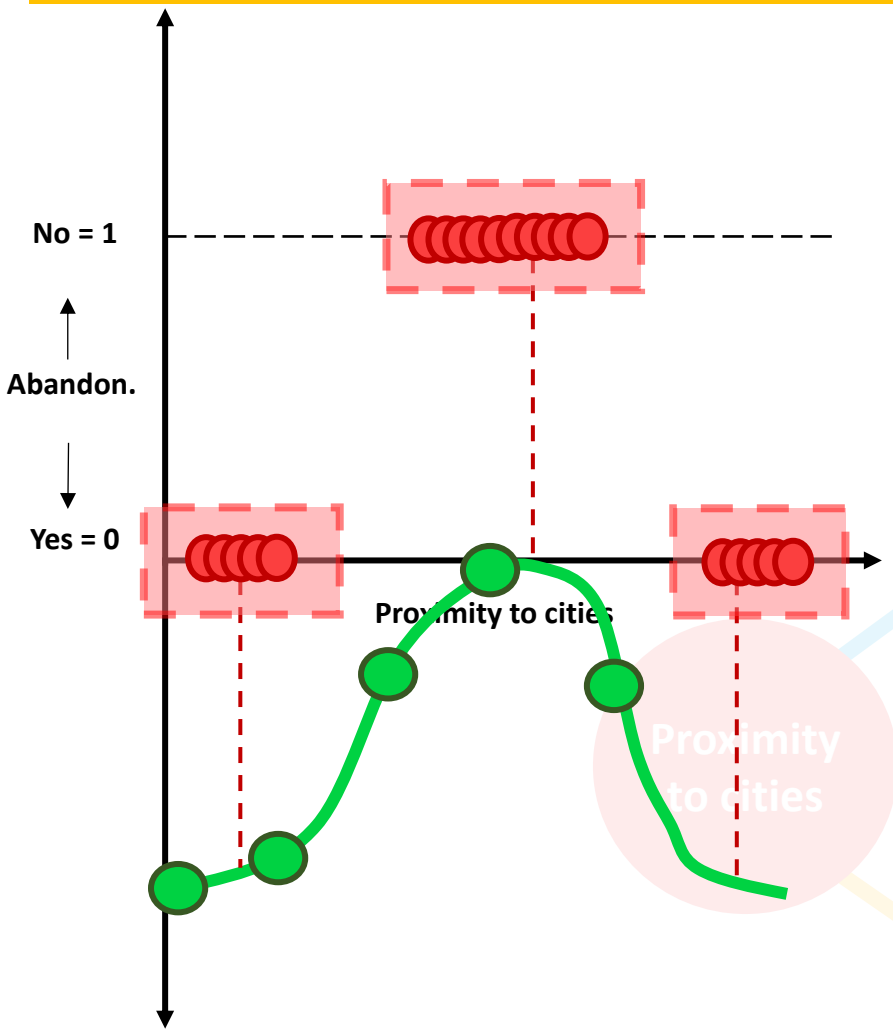
Observe que estamos nos aproximando dos valores de nossos <u>dados de treinamento</u> à medida que aumentamos o valor de b_3

b_3

Soma + 0.1 =

Em outras palavras, estamos <u>minimizando os resíduos</u> alterando o valor de b_3

Como estimar parâmetros em uma RNA?



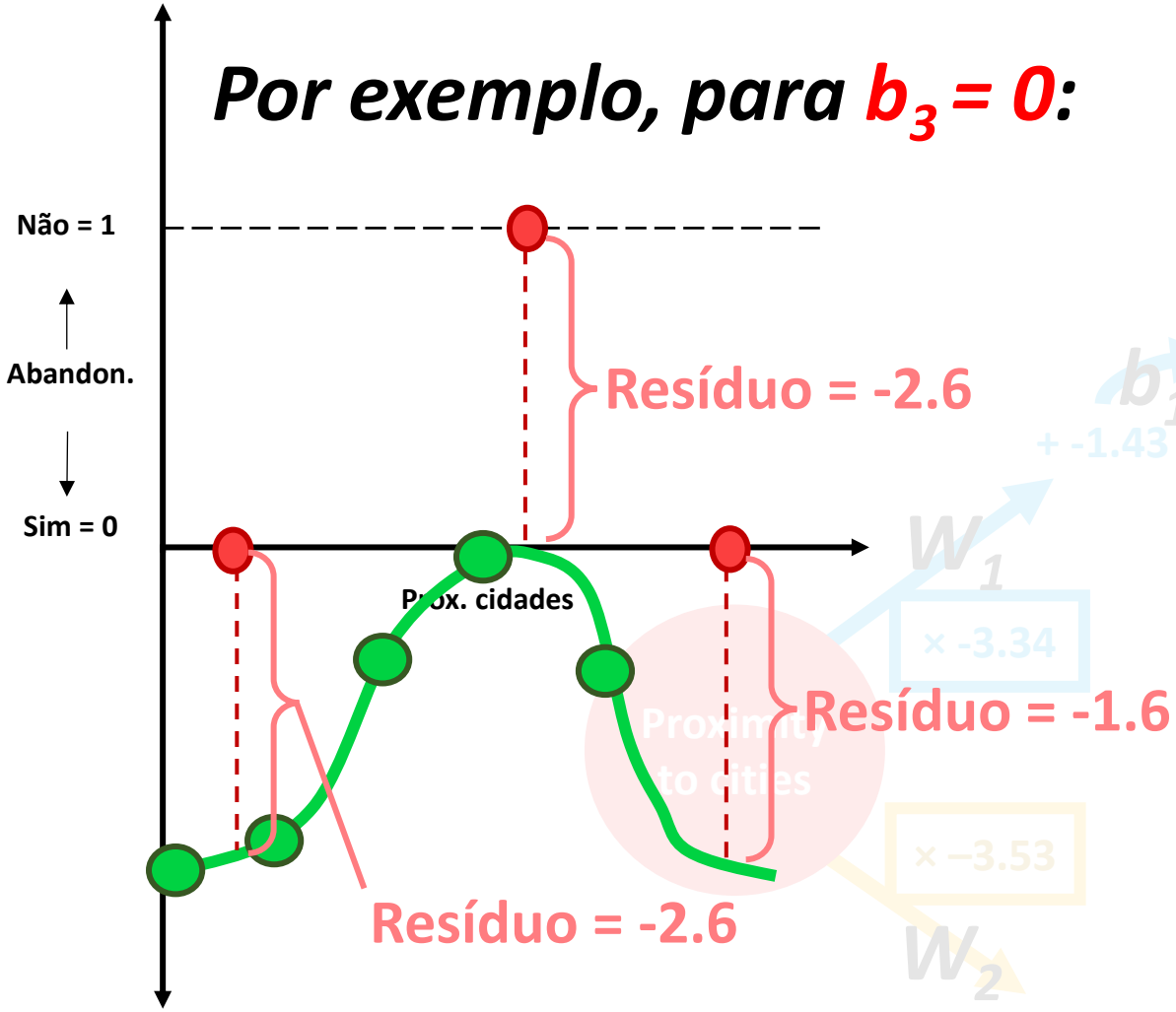
Lembre-se: temos tantos resíduos quanto o número de unidades de treinamento usadas para ajustar a RNA

A RNA tenta minimizar todos os resíduos, e não apenas como são, mas geralmente suas versões <u>ao quadrado</u>

Objetivo: encontrar o valor ótimo de b_3 que minimize a <u>Soma dos Resíduos Quadrados</u>

Como estimar parâmetros em uma RNA?

Por exemplo, para $b_3 = 0$:



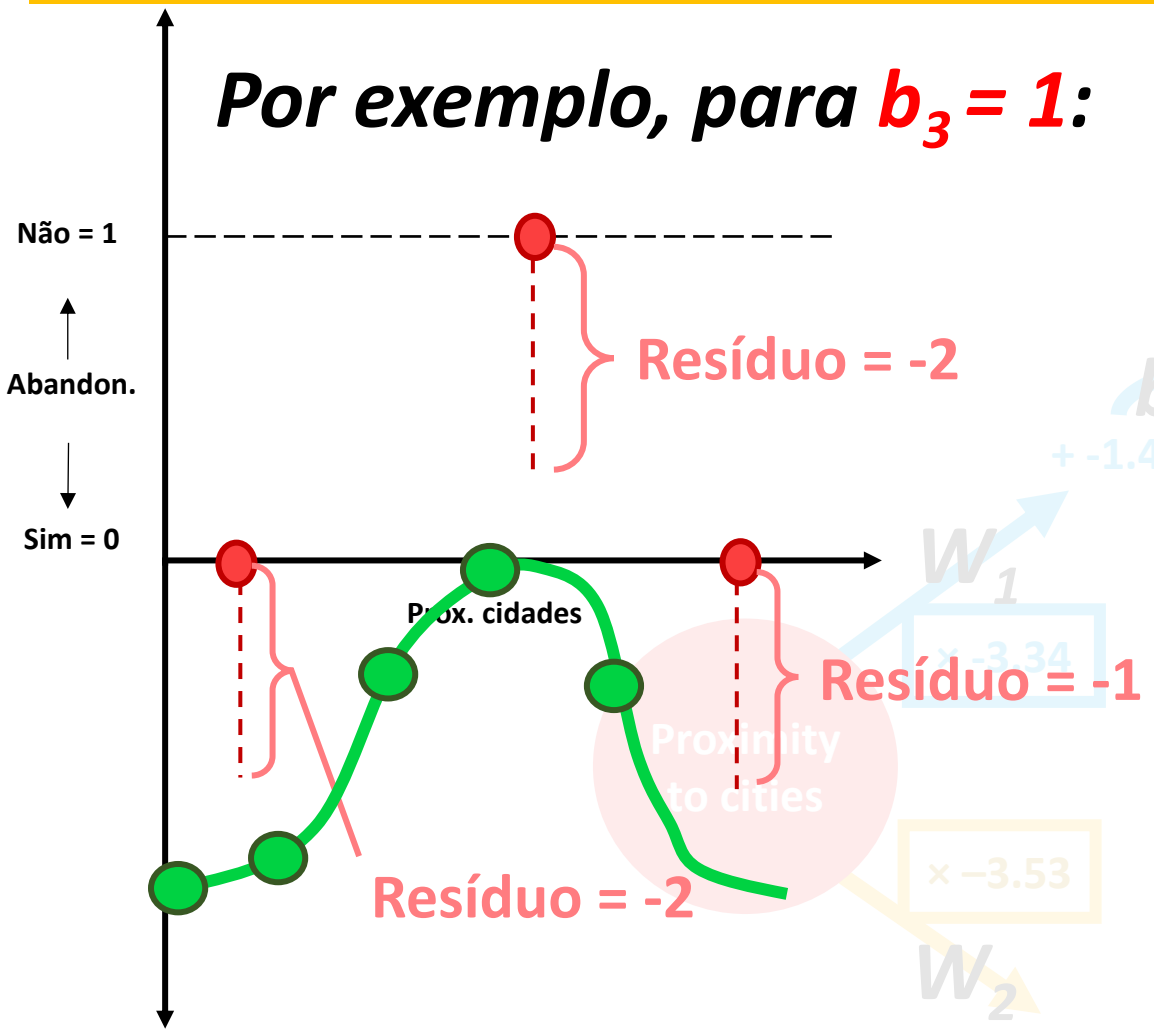
A <u>Soma dos Resíduos Quadrados</u> (SRQ) é:

$$SSR = (-2.6)^2 + (-1.6)^2 + (-2.6)^2$$
$$SSR = 20.4$$

… e podemos repetir isso para outros valores de b_3

Como estimar parâmetros em uma RNA?

Por exemplo, para $b_3 = 1$:

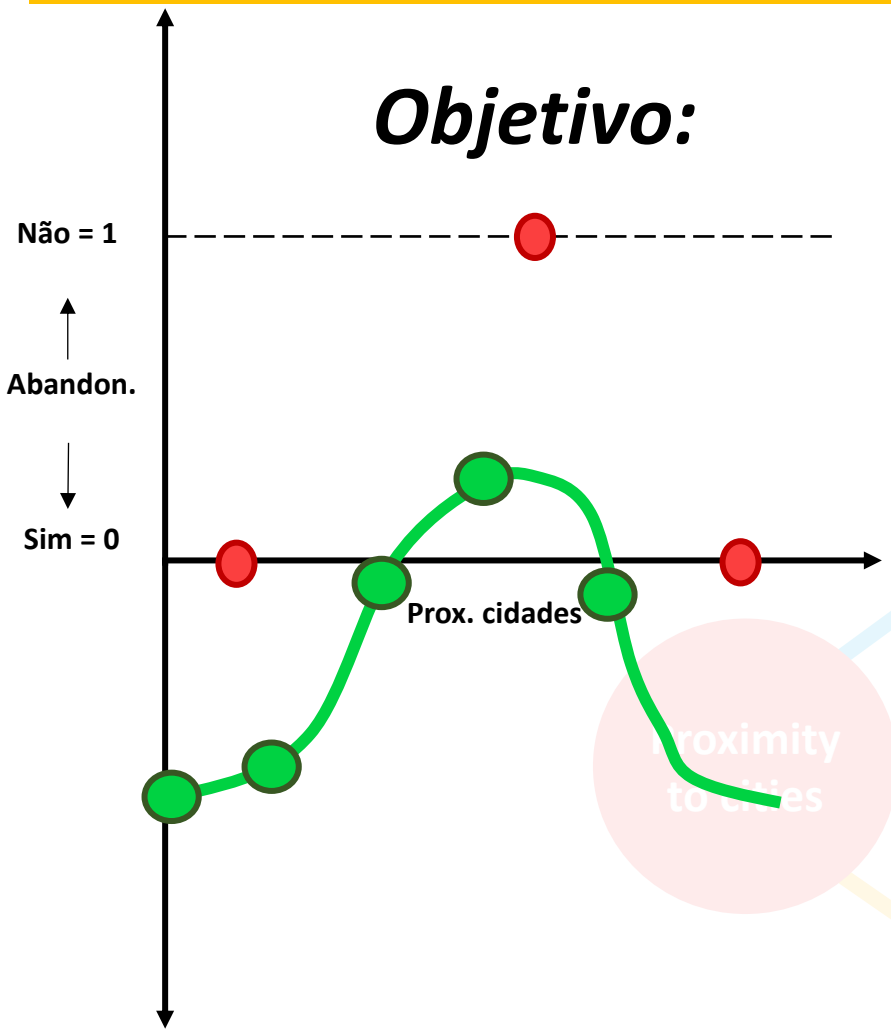


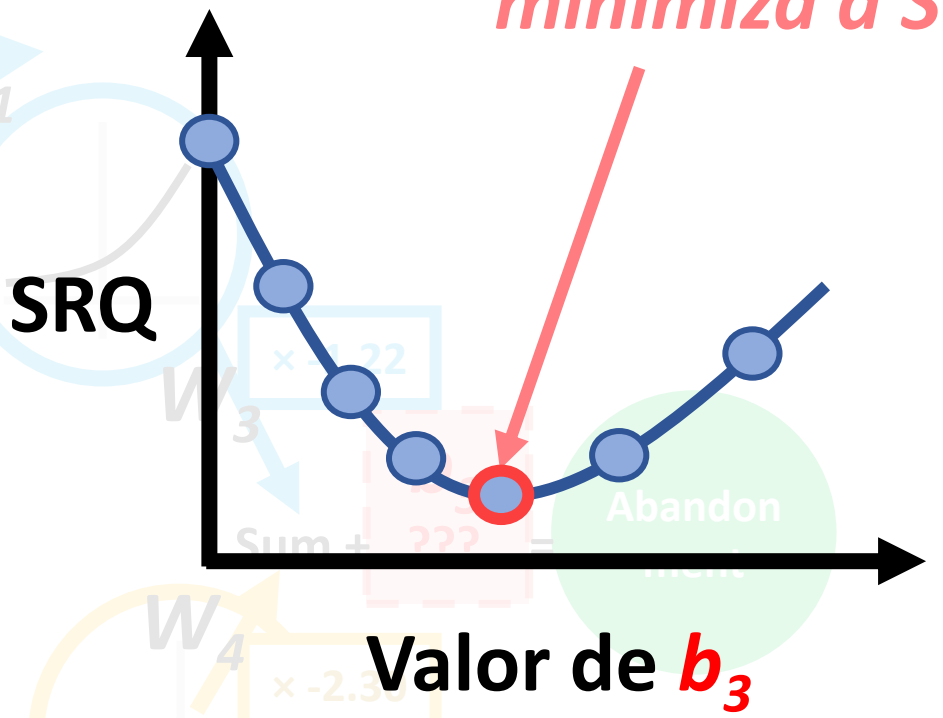
A <u>Soma dos Resíduos Quadrados</u> (SRQ) é:

$$SSR = (-2)^2 + (-1)^2 + (-2)^2$$

$$SSR = 5$$

Como estimar parâmetros em uma RNA?



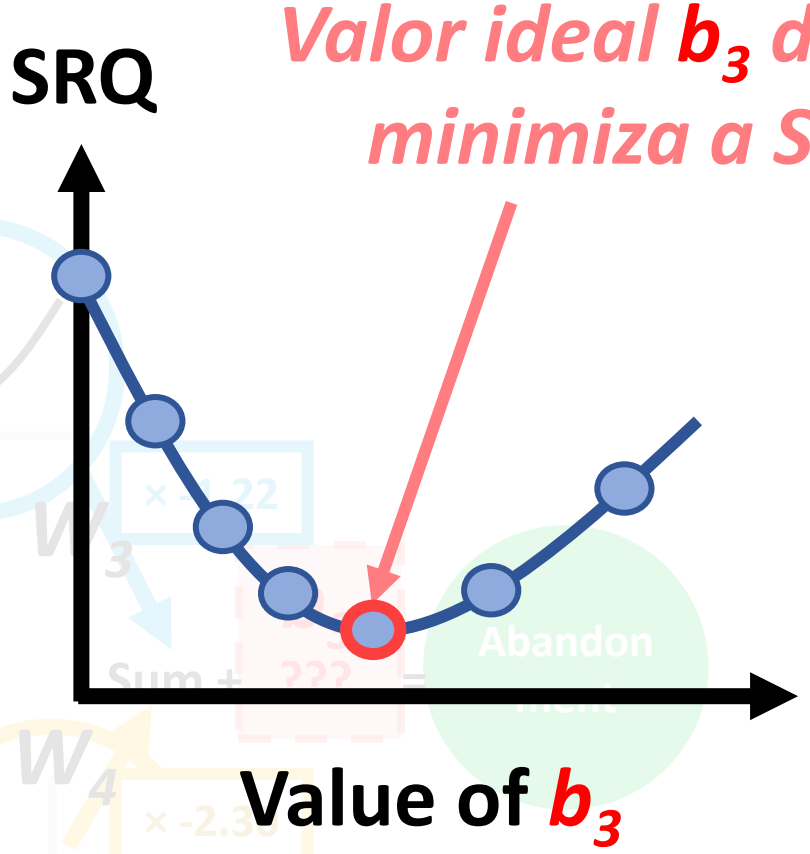


Como estimar parâmetros em uma RNA?

Podemos inserir diferentes valores de b_3 identificar seu valor ótimo, mas isso não é muito eficiente...

Em vez disso, as RNA usam métodos de otimização, como <u>Gradiente Descendente</u>

(Gradient Descent)



Como estimar parâmetros em uma RNA?

Podemos inserir diferentes valores de b_3 identificar seu valor ótimo, mas isso não é muito eficiente...

Em vez disso, as RNA usam métodos de otimização, como <u>Descida de Gradiente</u> ou <u>Gradiente Descendente</u>
(Gradient Descent)

$$\frac{\partial\, SSR}{\partial\, b_3} = \sim 0$$

Como estimar parâmetros em uma RNA?

$$\frac{\partial\, SSR}{\partial\, b_3} = \sim 0$$

$$b_{3_{New}} = b_{3_{Old}} - \eta \frac{\partial\, Loss}{\partial\, b_3}$$

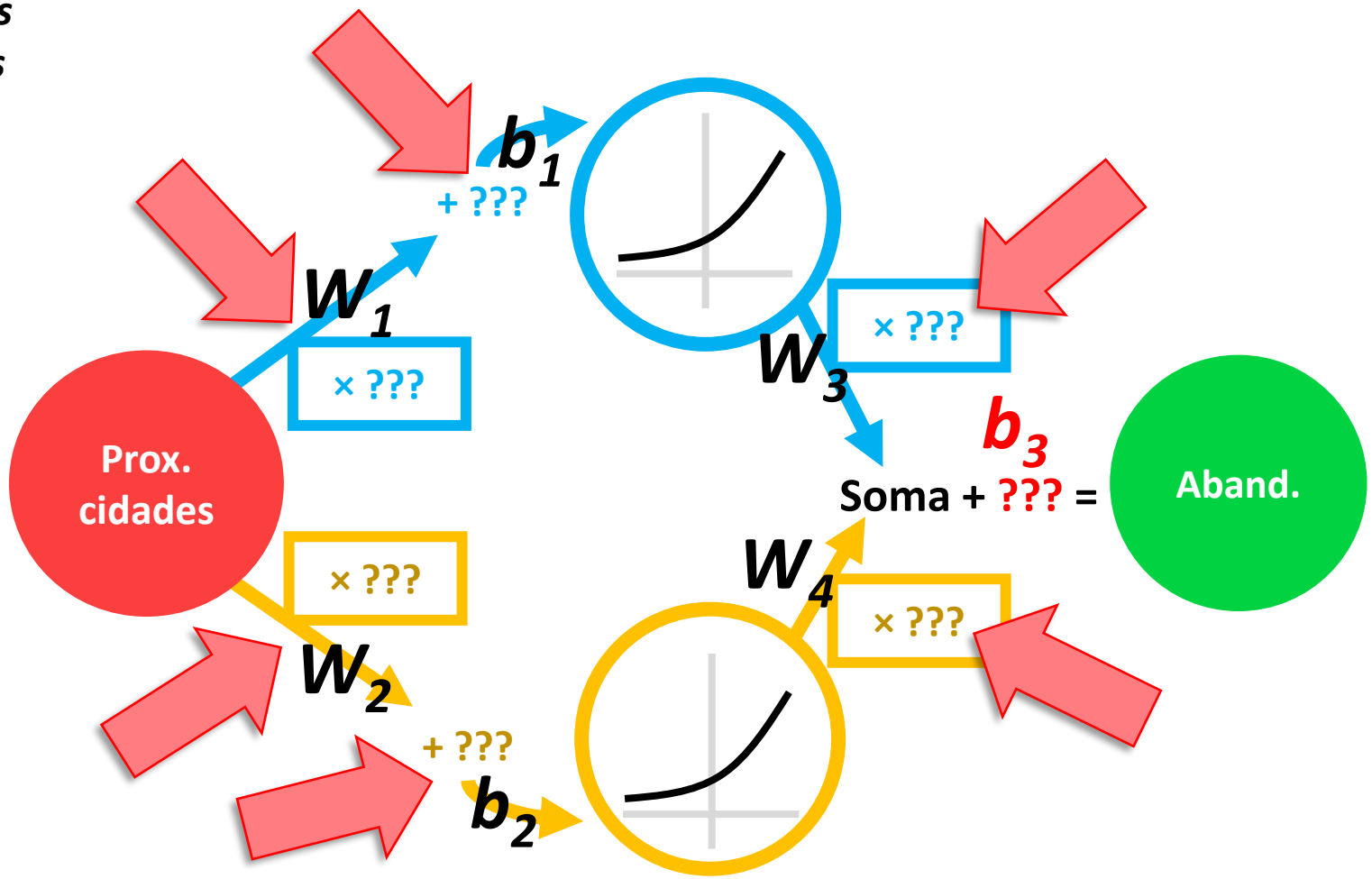
Note:
Loss = "Loss function"
η = Learning rate

E quanto aos outros parâmetros da RNA?

Antes que a RNA comece com as derivadas, precisamos inserir alguns valores aleatórios nesses parâmetros (geralmente o algoritmo da RNA faz isso por nós)

A RNA então começa a alterá-los um pouco com base nos resultados das derivadas

A quantidade de mudança é chamada de <u>Taxa de Aprendizado</u> (Learning Rate)



$$\frac{\partial\, SSR}{\partial\, b_3}$$

$$\frac{\partial\, SSR}{\partial\, W_3}$$

$$\frac{\partial\, SSR}{\partial\, W_4}$$

$$\vdots$$

Retropropagação com Descida de Gradiente

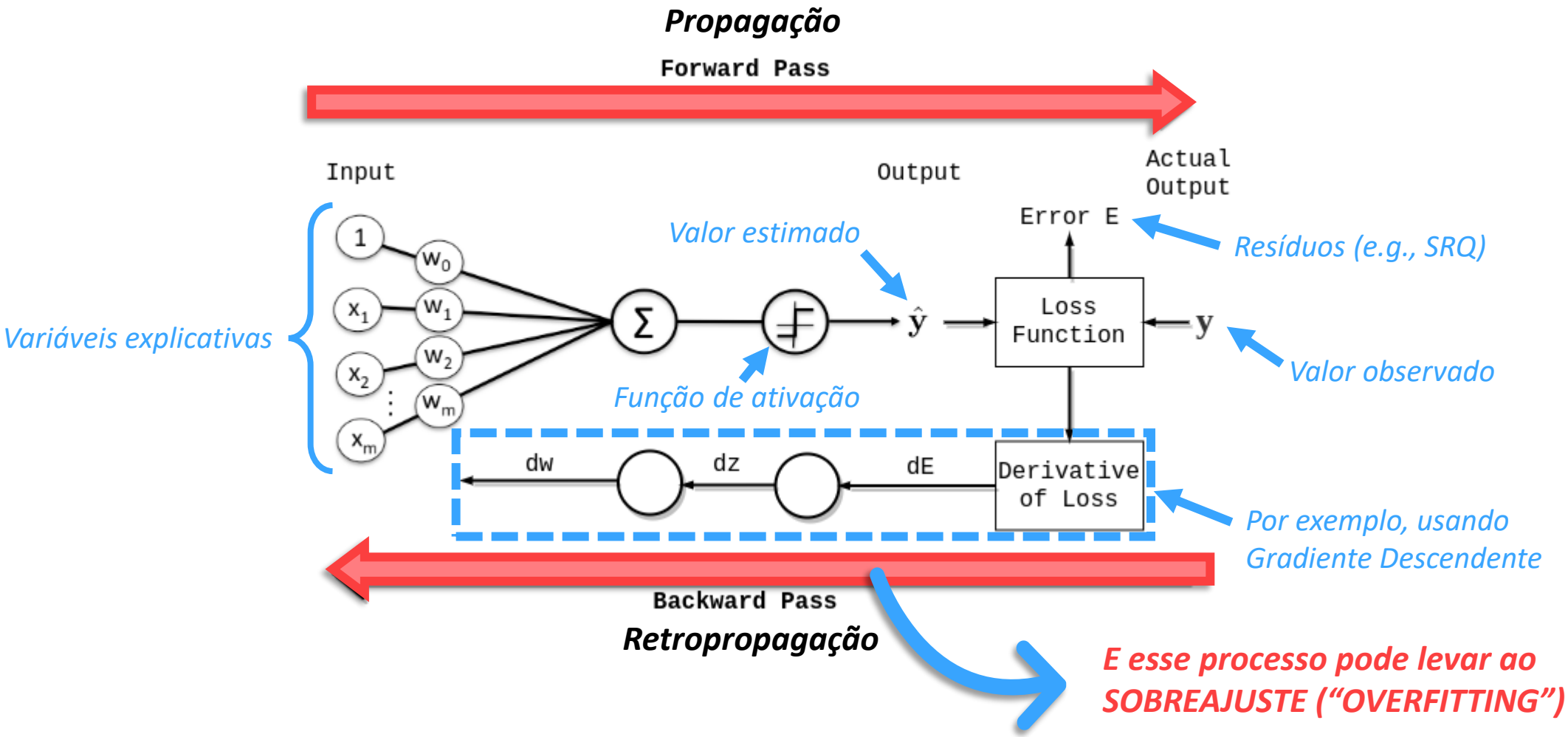
O ***Gradient Descent*** é um algoritmo de otimização usado para minimizar uma *função de custo* ("cost function")* em modelos de aprendizado de máquina, particularmente no treinamento de RNA. Em linhas gerais, essas são as etapas a serem seguidas:

1. ***Inicializar parâmetros:*** comece inicializando os parâmetros do modelo (weights e biases) aleatoriamente ou com alguns valores predefinidos.
2. ***Calcular previsões:*** use os valores de parâmetro atuais para fazer previsões sobre os dados de treinamento. Esta etapa envolve a propagação dos dados de entrada pela rede para produzir as saídas prevista.
3. ***Calcular perda/custo:*** calcule a *função de perda ou custo*, que mede a diferença entre as saídas previstas e os rótulos/valores reais. A escolha da *função de perda* depende do problema específico que está sendo resolvido (por exemplo, erro quadrático médio para regressão ou perda de entropia cruzada para classificação).
4. ***Calcular gradiente:*** calcule o gradiente da função de perda em relação a cada parâmetro do modelo. O gradiente representa a direção e a magnitude do aumento mais acentuado na função de perda. Esta etapa envolve a retropropagação do erro pela rede para calcular os gradientes de forma eficiente usando técnicas como a "regra da cadeia" (ou seja, com base em derivadas).
5. ***Atualizar parâmetros:*** atualize os parâmetros do modelo na direção oposta ao gradiente para minimizar a *função de perda*. Esta etapa envolve subtrair uma fração do gradiente dos valores atuais dos parâmetros, dimensionados pelo hiperparâmetro da taxa de aprendizado.
6. ***Repetir:*** Repita as etapas 2 a 5 para um número fixo de iterações (épocas) ou até que os critérios de convergência sejam atendidos. Os critérios de convergência podem incluir atingir um limite mínimo para a função de perda ou observar pouca melhoria na função de perda em iterações consecutivas.

Ao atualizar iterativamente os parâmetros do modelo com base no gradiente da *função de perda*, o ***Gradient Descent*** se move em direção ao conjunto ideal de parâmetros que minimizam a perda e melhoram o desempenho do modelo baseando-se nos dados de treinamento.

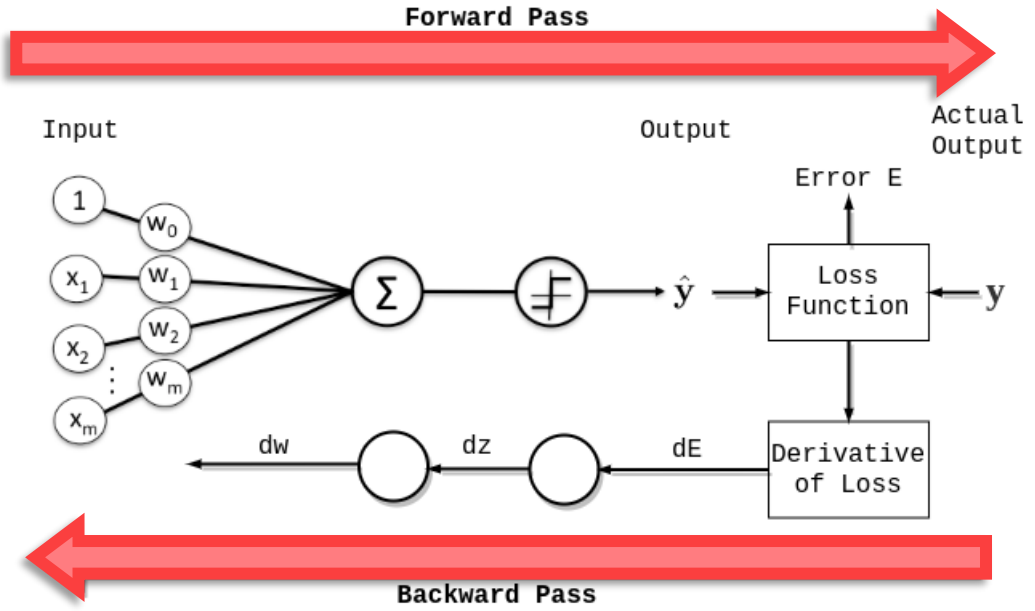
*Em RNA, uma *função de custo* [também conhecida como *função de perda ("loss function")* ou *função objetivo ("objective function")*] é uma medida de quão bem as previsões da RNA correspondem aos valores-alvo reais. O objetivo do treinamento de um RNA é minimizar essa *função de custo*, pois ela representa a discrepância entre as saídas previstas e os rótulos ou destinos verdadeiros.

Retropropagação em RNA



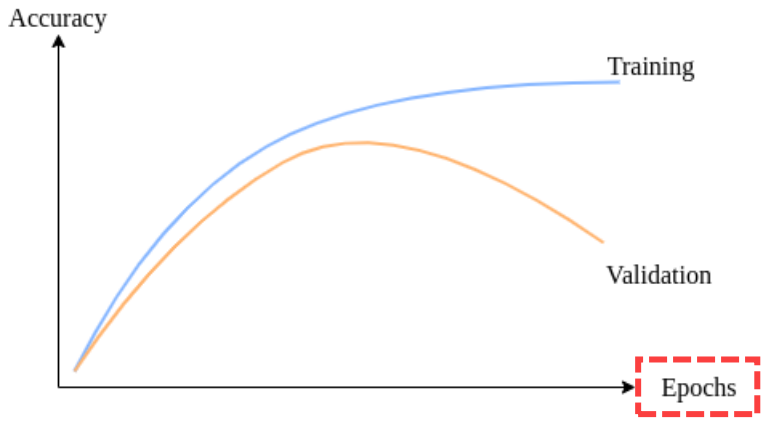
Retropropagação em RNA

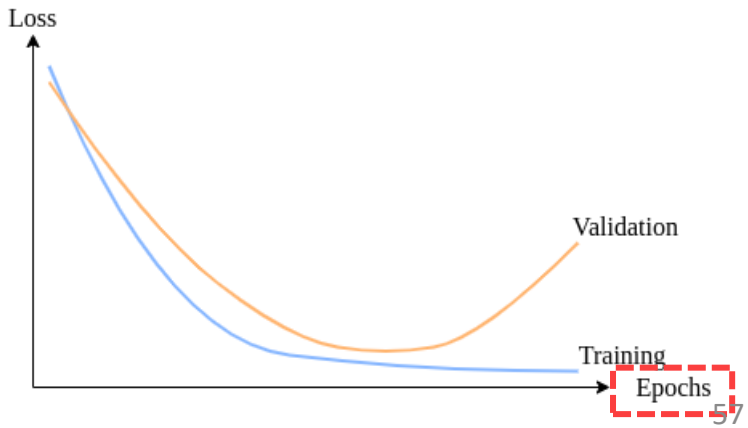
Em algoritmos de RNA, alguns dos dados (por exemplo, 20%) são usados para conduzir um "validação interna" do modelo de RNA à medida que o modelo é calibrado iterativamente [através de cada época ("epoch")]



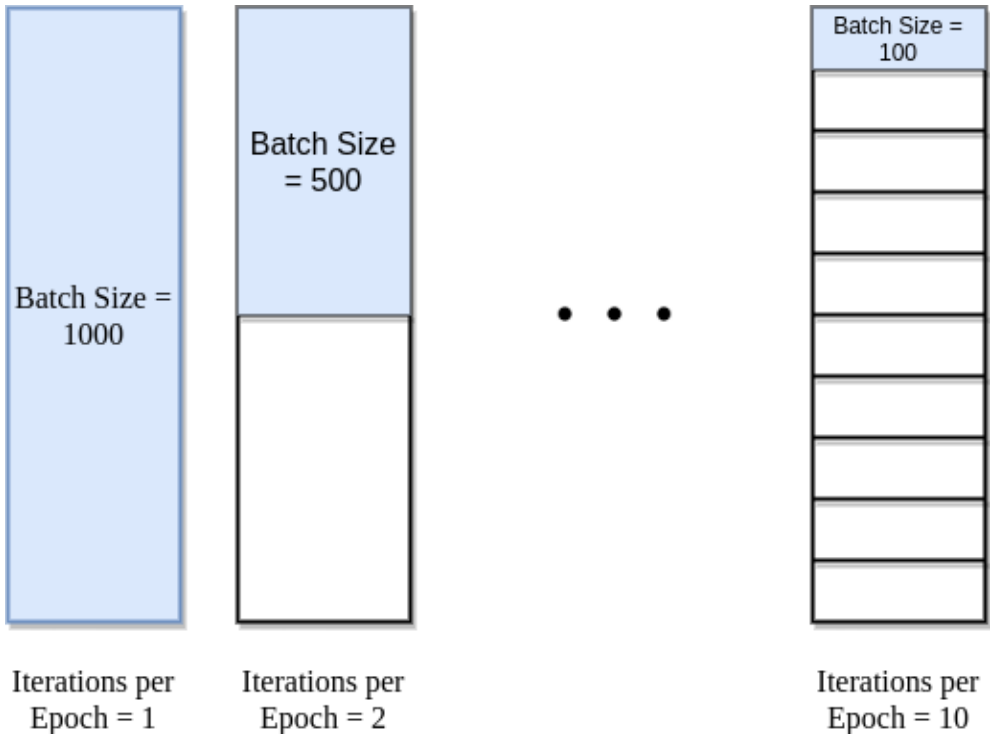
Época (Epoch) = uma passagem (ida e volta) dos <u>dados de treinamento</u> pela RNA

Isso significa que uma parte dos <u>dados de treinamento</u> são de fato utilizados como <u>dados de treinamento</u> e enquanto outra parte é utilizada como "dados de validação (interna)"





Retropropagação em RNA



Digamos que nossos dados contém 1000 observações:

→ Se o tamanho do lote (batch) for 1000, podemos completar uma época (epoch) com 1 única iteração

→ Se o tamanho do lote for 500, uma época leva 2 iterações

→ Se o tamanho do lote for 100, uma época leva 10 iterações

Para cada época (epoch), o número necessário de iterações vezes o tamanho do lote é igual ao número de observações no conjunto de dados de treinamento

"Lotes" (batches) são utilizados para:
- Superar as limitações de processamento (eficiência)
- Habilitar computação paralela
- Reduzir o sobreajuste
- Entre outras razões ...

Outras maneiras de evitar “overfitting”...

Técnicas de *regularização* adicionam restrições ao processo de aprendizado, normalmente adicionando penalidades à *função de perda* ou modificando diretamente os parâmetros do modelo. O objetivo é desencorajar modelos excessivamente complexos que se encaixam no “ruído” nos dados de treinamento, em vez das “relações verdadeiras” entre as variáveis/dados Alguns exemplos de *regularização* são:

Laço (“Lasso”): Adiciona uma penalidade proporcional ao valor absoluto dos pesos (weights) à função de perda. Ele incentiva a dispersão nos pesos, o que significa que alguns pesos se tornam exatamente zero.

Cume (“Ridge”): Adiciona uma penalidade proporcional à magnitude quadrada dos pesos (weights) à função de perda. Ele incentiva pesos menores e pode impedir que pesos grandes dominem o processo de otimização.

“Dropout”: Descarta aleatoriamente unidades (“nodes”) da RNA durante o treinamento, forçando a RNA a aprender representações redundantes dos dados. Isso ajuda a evitar o *sobreajuste*, tornando a rede mais robusta a variações nos dados de entrada.

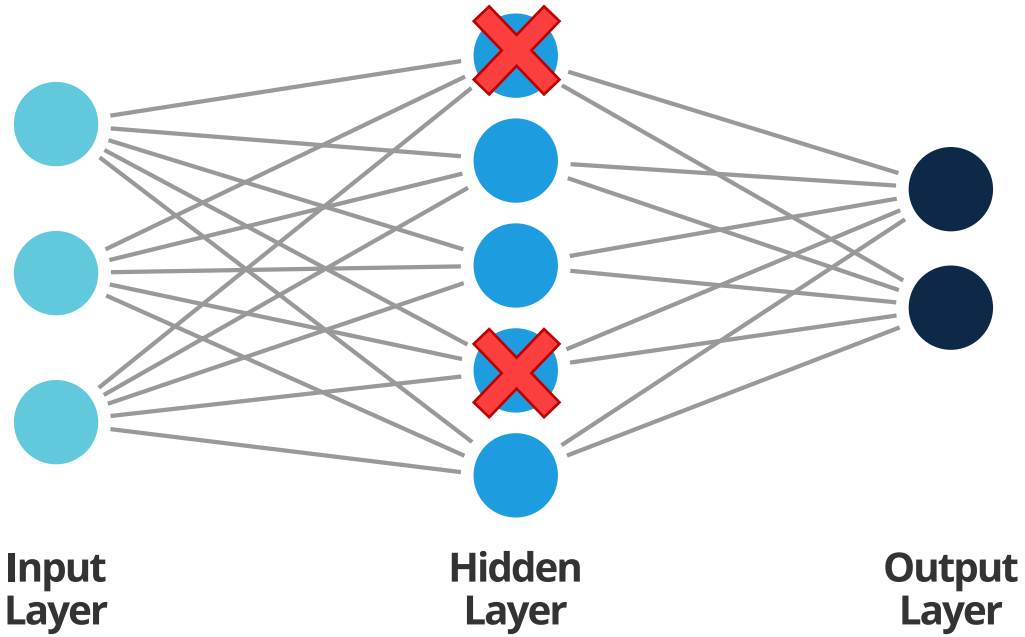
Normalização em lote (“Batch Normalization”): Normaliza os dados em cada camada na RNA (antes de passarem pela *função de ativação*) para ter média zero e variância unitária. Isso pode ajudar a estabilizar o processo de treinamento.

Parada antecipada (“Early Stopping”): monitora o desempenho do modelo *baseado na parte dos dados de treinamento utilizados para “validação interna”* e interrompe o treinamento quando o desempenho começa a diminuir. Isso evita que o modelo se ajuste demais aos dados de treinamento.

Essas técnicas de *regularização* podem ser aplicadas individualmente ou em combinação para controlar a complexidade do modelo e melhorar seu desempenho de generalização.

Outras maneiras de evitar “overfitting”: Dropout

Uma % dos neurônios da(s) hidden layer(s) é aleatóriamente descartada (= zero) a cada iteração



Resumo geral da otimização de RNA

- **Antes que a RNA inicie o processo de otimização, os weights e biases (parâmetros da rede) precisam ser inicializados com valores aleatórios. Geralmente, o próprio algoritmo da RNA faz essa inicialização automaticamente**

- **Após a inicialização, o algoritmo da RNA começa a ajustar esses valores iniciais com base nos resultados das derivadas da função de perda (e.g., usando *Gradient Descent*). O objetivo é minimizar o erro ou os resíduos, como a *Soma dos Resíduos Quadrados*, comparando as previsões do modelo com os valores reais (da parte dos dados de treinamento utilizados para "*validação interna*")**

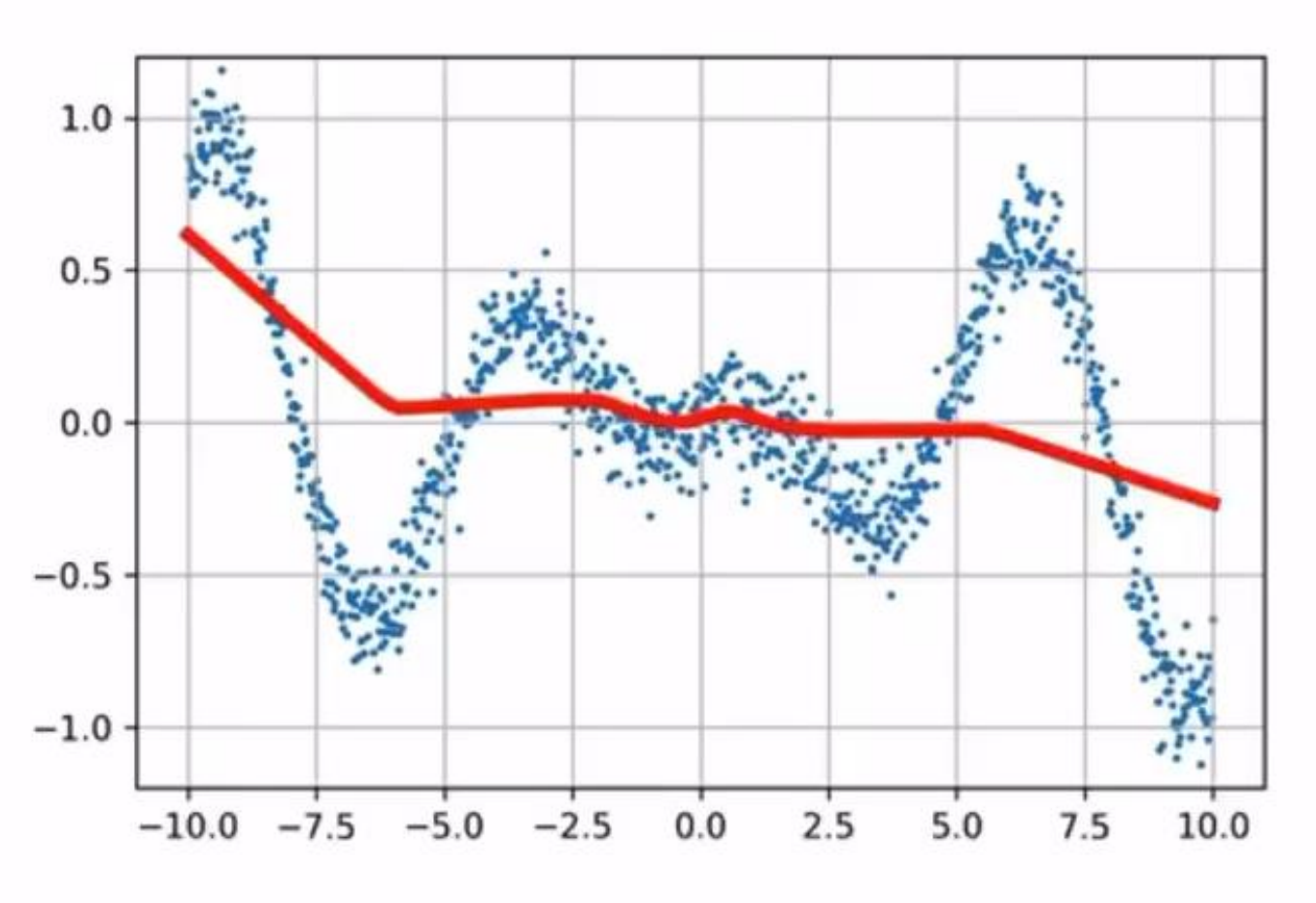
- **A velocidade desses ajustes é controlada pela *Taxa de Aprendizado*, que define o tamanho dos passos dados durante a optimização dos parâmetros. Uma taxa de aprendizado muito alta pode fazer com que a rede "pule" o ponto ótimo, enquanto uma taxa muito baixa pode tornar o processo de treinamento muito lento**

- **O processo de treinamento continua até que um *número máximo de iterações* (ou *épocas*) seja alcançado, ou até que o *erro atinja um nível aceitável*, conforme definido pelos parâmetros estabelecidos pelo usuário**

Vamos ver uma RNA em ação...

Pontos azuis ***são os*** ***dados de treinamento*** ***da RNA***

A linha vermelha é o "rabisco" que a RNA está ajustando aos dados, o qual pode ser utilizado posteriormente para fazer previsões



Resumo

- ***RNA podem ser usadas para estabelecer relações não lineares altamente complexas entre variáveis***
- ***As funções de ativação estão no centro dos modelos de RNA***
- ***Os parâmetros (weights e biases) são estimados iterativamente com base em retropropagação***
- ***RNA podem facilmente se deparar com problemas de sobreajuste (overfitting), principalmente se ignorarmos como a precisão/perda do modelo muda ao longo das iterações/épocas***